RESEARCH IN COMPUTING SCIENCE

ISSN: 1870-4069

Special issue: Natural Language Processing and its Applications

Alexander Gelbukh
(Ed.)

Vol. 46

Special issue: Natural Language Processing and its Applications

Research in Computing Science

Research in Computing Science es una publicación trimestral, de circulación internacional, editada por el Centro de Investigación en Computación del IPN, para dar a conocer los avances de investigación científica y desarrollo tecnológico de la comunidad científica internacional. **Volumen 46** Marzo, 2010. Tiraje: 500 ejemplares. *Certificado de Reserva de Derechos al Uso Exclusivo del Título* No. 04-2004-062613250000-102, expedido por el Instituto Nacional de Derecho de Autor. *Certificado de Licitud de Título* No. 12897, *Certificado de licitud de Contenido* No. 10470, expedidos por la Comisión Calificadora de Publicaciones y Revistas Ilustradas. El contenido de los artículos es responsabilidad exclusiva de sus respectivos autores. Queda prohibida la reproducción total o parcial, por cualquier medio, sin el permiso expreso del editor, excepto para uso personal o de estudio haciendo cita explícita en la primera página de cada documento. Impreso en la Ciudad de México, en los Talleres Gráficos del IPN – Dirección de Publicaciones, Tres Guerras 27, Centro Histórico, México, D.F. Distribuida por el Centro de Investigación en Computación, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othón de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, México, D.F. Tel. 57 29 60 00, ext. 56571.

Editor Responsable: *Juan Humberto Sossa Azuela, RFC SOAJ560723*

Volume 46

Volumen 46

Special issue:
Natural Language Processing and its Applications

Volume Editor:
Editor de Volumen

Alexander Gelbukh

Instituto Politécnico Nacional
Centro de Investigación en Computación
México 2010

ISSN: 1870-4069

Instituto Politécnico Nacional (IPN)
Centro de Investigación en Computación (CIC)
Av. Juan de Dios Bátiz s/n esq. M. Othón de Mendizábal
Unidad Profesional "Adolfo López Mateos", Zacatenco
07738, México D.F., México

http://www.ipn.mx
http://www.cic.ipn.mx

Indexed in LATINDEX and Periodica / Indexada en LATINDEX y Periodica

Printing: 500 / Tiraje: 500

Printed in Mexico / Impreso en México

Preface

Natural Language Processing is an interdisciplinary research area at the border between linguistics and artificial intelligence aiming at developing computer programs capable of human-like activities related to understanding or producing texts or speech in a natural language, such as English or Chinese.

The most important applications of natural language processing include information retrieval and information organization, machine translation, and natural language interfaces, among others. However, as in any science, the activities of the researchers are mostly concentrated on its internal art and craft, that is, on the solution of the problems arising in analysis or generation of natural language text or speech, such as syntactic and semantic analysis, disambiguation, or compilation of dictionaries and grammars necessary for such analysis.

This volume presents 27 original research papers written by 63 authors representing 25 different countries: Argentina, Canada, China, Cuba, Czech Republic, Denmark, France, Germany, India, Indonesia, Islamic Republic of Iran, Italy, Japan, Republic of Korea, Mexico, Republic of Moldova, Pakistan, Portugal, Romania, Spain, Sweden, Tajikistan, Turkey, United Kingdom, and United States. The volume is structured in 8 thematic areas of both theory and applications of natural language processing:

- Semantics
- Morphology, Syntax, Named Entity Recognition
- Opinion, Emotions, Textual Entailment
- Text and Speech Generation
- Machine Translation
- Information Retrieval and Text Clustering
- Educational Applications
- Applications

The papers included in this volume were selected on the base of rigorous international reviewing process out of 101 submissions considered for evaluation; thus the acceptance rate of this volume was 27%.

I would like to cordially thank all people involved in the preparation of this volume. In the first place I want to thank the authors of the published paper for their excellent research work that gives sense to the work of all other people involved, as well as the authors of rejected papers for their interest and effort. I also thank the members of the Editorial Board of the volume and additional reviewers for their hard work on reviewing and selecting the papers. I thank Sulema Torres and Corina Forăscu for their valuable collaboration in preparation of this volume. The submission, reviewing, and selection process was supported for free by the EasyChair system, www.EasyChair.org.

Alexander Gelbukh March 2010

Table of Contents

Semantics

Morphology, Syntax, Named Entity Recognition

Opinion, Emotions, Textual Entailment

Text and Speech Generation

Machine Translation

Information Retrieval and Text Clustering

Educational Applications

Applications

Semantics

Lexical Representation of Agentive Nominal Compounds in French and Swedish

Maria Rosenberg

Stockholm University, maria.rosenberg @fraita.su.se

Abstract. This study addresses the lexical representation of French VN and Swedish NV-*are* agentive nominal compounds. The objective is to examine their semantic structure and output meaning. The analysis shows that, as a result of their semantic structure, the compounds group into some major output meanings. Most frequently, the N constituent corresponds to an Undergoer in the argument structure of the V constituent, and the compound displays an Actor role, which more precisely denote entities such as Persons, Animals, Plants, Impersonals, Instruments or Locatives, specified in the Telic role in the Qualia. We propose that the Agentive role can be left unspecified with regard to action modality. In conclusion, our study proposes a unified semantic account of the French and Swedish compounds, which can have applications for NLP systems, particularly for disambiguation and machine translation tasks.

Keywords. Agentive nominal compounds, Actor, Undergoer, semantic structure, lexical representation, Generative Lexicon, telic, disambiguation

1 Introduction

This study addresses the semantics of French and Swedish agentive nominal compounds that contain an N and a V constituent, manifesting an argumental relation. French has only one such compound type, which has exocentric structure [1][1]:

- $[VN_y]_{Nx}$: *porte-drapeau* 'bear-flag=flag bearer'

Table 1 shows the initial data for our study, which aimed to localize Swedish correspondents of French VN compounds. By going through four bilingual French-Swedish dictionaries, we attested 432 French nominal VN compounds. Among these, 229 were rendered by four Swedish compound types. The remaining cases corresponded mainly to simple words or syntactic phrases. Apart from the data in Table 1, our data draws mainly from dictionaries (*TLFi* and *SAOB*) and the Internet. We support our analysis by a restricted sample of representative examples.

[1] Romance VN compounds are also analyzed as left-headed: a nominal zero suffix adds to the V [2], or as right-headed: a nominal zero suffix adds to the compound, considered as a VP, [3].

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 3-15

Received 24/11/09
Accepted 16/01/10
Final version 08/03/10

Table 1. French VN compounds and their corresponding Swedish compounds in four bilingual dictionaries.

Compound type	*n*
VN (fra)	432
NV-*are* (swe)	108
NV-*a* (swe)	16
NV (swe)	54
VN (swe)	51

According to Table 1, the Swedish NV-*are* compound is the most frequent counterpart. Hence, we focus solely on this Swedish compound in this study. Swedish NV-*are* compounds are right-headed and also called synthetic, since they involve both compounding and derivation. Their formation can be considered as a process of conflation, uniting two templates $[NV]_V$ and $[V\ are]_N$ into a unified productive template $[[NV]\ are]_N$ (cf. [1]):

- $[[NV]\ are]_N$: *fanbärare* 'flag bearer'

French VN and Swedish NV-*are* compounds give rise to polysemy and sense shifting within the agentive domain. They do not only denote humans, but all sorts of animate entities, e.g. animals, plants and insects, as well as artefacts, e.g. instruments, impersonal agents and locatives. They can also refer to places, events and results. Thus, morphologically, VN and NV-*are* correspond to two constructions, which have several underlying semantic structures.

Our main objective is to examine the lexical representation of the compounds. We explore the semantic roles of the N constituents and the semantic characteristics of the V constituents, as well as their semantic structures. Note that both the N constituent and the compound fulfil different roles in the argument structure of the V. The role displayed by the compound corresponds to its output meaning. Despite the formal differences of French and Swedish compounds, we aim at a unified semantic account. Moreover, we discuss the importance of action modality as a component in the lexical representation of agentive compounds. Automatic analyses of nominal compounds constitute an intriguing question within NLP. In order for our study to have some predictive power, we attempt to relate the semantic structures and output meanings to productivity and frequency. At the present, we are setting the ground for a future implementation of regular lexical morphology principles in a machine translation (MT) system.

Section 2 addresses the morphological context. Section 3 deals with the semantic characteristics of the constituents within the compounds. In section 4, we analyze the semantic structures and the output meanings of the compounds. Section 5 discusses the notion of action modality. In Section 6, we propose GL representations for the three most frequent cases. Section 7 discusses potential applications within NLP, and Section 8 contains a conclusion.

2 Compounds in Morphology

The morphological approach is lexeme-based, and adheres to Construction Morphology, being elaborated by Booij (e.g. [1]). A compound is defined as a sequence which cannot be generated otherwise than by morphological rules. Hence, it does not have syntactic structure [4]. This criterion is valid for French VN and Swedish NV-*are* compounds. Interaction between syntax and lexicon is however tolerated: the lexical rules may make use of syntactic information [5]. "Word-formation patterns can be seen as abstract schemas that generalize over sets of existing complex words with a systematic correlation between form and meaning" [1]. The generalizations are expressed in the lexicon by assuming intermediate levels of abstractions between the most general schema and individual existent compounds. Hence, the lexicon is hierarchically structured. The morphology combines three aspects of complex words: phonological form, formal structure and meaning. The architecture of grammar is tripartite and parallel [6]. Abstract schemas coexist with their individual instantiations in the lexicon. Thus, outputs of productive rules can also be listed [1].

3 Semantic Characteristics of the Compounds

3.1 Argument Structure and Semantic Roles of the N Constituent

Agentive compounds contain an argumental relation between the V and N constituents. According to our analysis, the N constituent can correspond, more or less, to all four types of arguments, distinguished in the Generative Lexicon (GL) framework [7]:

- **True arguments**: *ouvre-boîte* 'open-can=can opener', *burköppnare* 'can opener'.
- **Default arguments**: *claque-soif* 'die-thirst=person dying of thirst', *cuit-vapeur* 'boil-steam=steamer', *ångkokare*, 'steam+boiler=steamer' *betonggjutare* 'concrete caster'.
- **Shadow arguments**: *marche-pied* 'march-foot=step, running board', *bensparkare* 'leg kicker'.
- **True adjuncts**: *réveille-matin* 'wake-morning=alarm clock', *trädkrypare* 'tree+crawler=bird'.

In other terms, the semantic roles of the N constituents can correspond to Agent (*croque-madame* 'crunch-madam=toast'), Patient[2] (*ouvre-boîte* 'open-can=can opener'), Theme (*hatthängare* 'hathanger=hat-rack'), Place (*bordslöpare* 'table+runner=cloth'), Time (*dagdrömmare* 'day dreamer'), Instrument/Manner (*cuit-vapeur* 'steam+boiler=steamer', *fotvandrare* 'footrambler', Cause (*claque-soif* 'die-thirst=someone dying of thirst', *sorgedrickare* 'grief+drinker') or Goal (*cherche-pain* 'search-bread=beggar', *målsökare* 'target seeker').

[2] Patient corresponds to an entity, internally affected by the event expressed by the V, whereas Theme corresponds to an entity in motion, in change or being located [8].

However, an overwhelming majority of the French VN, 96 % (415/432), and the Swedish NV-*are* compounds, 97% (105/108), in our data, contain an N which is a direct object of a transitive V. Furthermore, about 73 % (79/108) of the Swedish NV-*are* counterparts of the French VN compounds contain semantically similar lexical units, such as *allume-gaz* 'light-gas=gas lighter' vs. *gaständare* 'gas lighter'. Hence, a MT system could benefit from an implementation of these facts (cf. section 7)

3.2 The Four Classes of *Aktionsart*

The four *Aktionsarten* [9] can occur within the French and Swedish compounds. The state reading is rare, but not unproductive. New formations arise quite easily, such as *godisälskare* 'candy lover'.

- **Activity**: *traîne-nuit* 'loaf-night', *dagdrivare* 'day loafer'.
- **Accomplishment**: *presse-citron* 'squeeze-lemon', *pennvässare* 'pencil sharpener'.
- **Achievement**: *presse-bouton* 'push-button', *cigarrtändare* 'cigar lighter'.
- **State**: *songe-malice* 'think-malice=someone who plots to evil', *vinkännare* 'wine+knower=connoisseur of wine'.

3.3 Unaccusative and Unergative Verbs

We see that both unaccusative and unergative readings of intransitive verbs can be attested within the compounds [10], [11]:

- **Unaccusative**: *caille-lait* 'clot-milk=plant', *oljedroppare* 'oil dripper'.
- **Unergative**: *trotte-bébé* 'baby walker', *hundpromenerare* 'dog+walker=person who takes the dog out for a walk'.

4 Semantic Structures and Output Meanings of the Compounds

The output meaning of French VN and Swedish NV-*are* compounds is taken to be a function of the meanings of their constituents [6]. The agentive compounds themselves display a role in the argument structure of the V (cf. [12] for French VN compounds). The N constituent can be classified for semantic macro-role, Actor or Undergoer [8] (correspond more or less to Proto-Agent and Proto-Patient [13]). In general, the compound corresponds to the Actor (including thematic roles such as Agent, Instrument and Experiencer), and its N constituent to an Undergoer (comprising roles such as Patient, Theme, Source and Recipient) of a transitive V. In order to come up with a more fine-grained semantic analysis, we split up the Actor interpretation into Actors corresponding to first arguments, Instruments, Locatives and Causatives.

We propose that the construction itself links to the output meaning (N_3). The structure of French VN compounds corresponds to $[V_1N_2]_{N3}$ in our analysis. The same proposal is made for Swedish NV-*are* compounds. Instead of linking the Actor

interpretation to the suffix, it links to the entire construction $[N_1V_2\text{-are}]_{N3}$. Thus, the meaning of the compound is the output of its semantic structure. We assume French VN and Swedish NV-*are* compounds to have similar semantic structures, their exocentricity or endocentricity, as well as the order between the V and N constituents, are of minor importance. Two implications follow from our proposal: the formal structure, not the *-are* suffix, is polysemous; null elements are not stipulated. We adopt Jackendoff's framework [6] to account for the semantic structures of the compounds. Table 2 shows the frequency of the output meanings of French VN and Swedish NV-*are* compounds in the initial data. The most frequent cases are Instrument, Agent and Instrumental Locative. They account for more than 90 % of all cases. This figure is confirmed for a collection of 1075 French VN compounds drawn from *TLFi* [14], but further data needs to be added for Swedish NV-*are* compounds.

Table 2. Output meanings of French VN and Swedish NV-*are* compounds in the initial data.

	ACTOR				UND	PLACE	EV	RES	*n*
	Arg1	INSTR	LOC	CAUS					
VN (fra)	128	193	84	3	2	1	18	3	432
	30%	45%	19%	0.7%	0.5%	0.2%	4%	0.7%	
NV-*are* (swe)	40	47	20	1					108
	37%	44%	19%	0.9%					

4.1 Actor is the First Argument

In the Actor interpretation, where the compound corresponds to the first argument of the V, we find compounds such as *porte-drapeau* or *fanbärare*, both 'flag bearer': 'a flag bearer bears a flag' (cf. 1-2). In some cases, the V is intransitive, and the N constituent displays a Place role (cf. 3-4). The compounds denote not only human agents, but also animals (cf. 3), plants, impersonals (cf. 5) (cf. also [15] who relates the Agent polysemy to the Animacy hierarchy). Sometimes, according to the semantics of the V, the compounds manifest an Experiencer role (cf. 6). According to [6], the function PROTECT (X, Y FROM Z) creates two groups of compounds, 'N2 protects N1' (cf. 7-8) and 'N2 protects from N1' (cf. 9-10), which denote some sort of disposal. According to Lieber "verbs which take more than one obligatory internal argument (e.g., *put*) [i.e. ditransitives] cannot form the base of synthetic compounds" [16]. This claim does not seem to be an absolute restriction, in any case not for French and Swedish (cf. also 17-18 in sub-section 4.3).

1. $[\text{porte-}_1\text{drapeau}_2]_3 = \text{PERSON}_3^{\alpha}$; $[\text{BEAR}_1\ (\alpha, \text{FLAG}_2)]$
2. $[\text{fan}_1\text{bär}_2\text{are}]_3 = \text{PERSON}_3^{\alpha}$; $[\text{BEAR}_2\ (\alpha, \text{FLAG}_1)]$
3. $[\text{trotte-}_1\text{chemin}_2]_3 = \text{ANIMAL}_3^{\alpha}$; $[\text{TROT}_1\ (\alpha, \text{ON ROAD}_2)]$
4. $[\text{kåk}_1\text{far}_2\text{are}]_3 = \text{PERSON}_3^{\alpha}$; $[\text{GO}_2\ (\alpha, \text{IN SLAMMER}_1)]$
5. $[\text{lave-}_1\text{vaisselle}_2]_3 = \text{MACHINE}_3^{\alpha}$; $[\text{WASH}_1\ (\alpha, \text{DISH}_2)]$
6. $[\text{vin}_1\text{känn}_2\text{are}]_3 = \text{PERSON}_3^{\alpha}$; $[\text{KNOW}_2\ (\alpha, \text{WINE}_1)]$
7. $[\text{garde-}_1\text{roue}_2]_3 = \text{DISPOSAL}_3^{\alpha}$; $[\text{PROTECT}_1\ (\alpha, \text{WHEEL}_2, \text{FROM INDEF})]$
8. $[\text{blus}_1\text{skydd}_2\text{are}]_3 = \text{DISPOSAL}_3^{\alpha}$; $[\text{PROTECT}_2\ (\alpha, \text{BLOUSE}_1, \text{FROM INDEF})]$
9. $[\text{garde-}_1\text{boue}_2]_3 = \text{DISPOSAL}_3^{\alpha}$; $[\text{PROTECT}_1\ (\alpha, \text{INDEF}, \text{FROM MUD}_2)]$

10. [blixt$_1$skydd$_2$are]$_3$ = DISPOSAL$_3^{\alpha}$; [PROTECT$_2$ (α, INDEF, FROM LIGHTNING$_1$)]

4.2 Instrument

Some Instrument denoting compounds are *ouvre-boîte* or *burköppnare*, both 'can opener': 'one opens a can with a can opener', or *casse-noix* or *nötknäppare*, both 'nutcracker'. This meaning is the most productive one in both French and Swedish.

11. [ouvre-$_1$boîte$_2$]$_3$ = INSTR$_3^{\alpha}$; [OPEN$_1$ (INDEF, CAN$_2$, WITH α)]
12. [burk$_1$öppn$_2$are]$_3$ = INSTR$_3^{\alpha}$; [OPEN$_2$ (INDEF, CAN$_1$, WITH α)]
13. [casse-$_1$noix$_2$]$_3$ = INSTR$_3^{\alpha}$; [CRACK$_1$ (INDEF, NUT$_2$, WITH α)]
14. [nöt$_1$knäpp$_2$are]$_3$ = INSTR$_3^{\alpha}$; [CRACK$_2$ (INDEF, NUT$_1$, WITH α)]

4.3 Locative

French VN and Swedish NV-*are* compounds do quite frequently have a Locative interpretation. It is close to the Instrument meaning, but instead of denoting something that one does things with, the compound denotes a location: 'one burns incense in a *brûle-parfum* 'censer'' (cf. 15-16) or 'one hangs saucepans on a saucepan hanger' (cf. 17-18).

15. [brûle-$_1$parfum$_2$]$_3$ = LOC$_3^{\alpha}$; [BURN$_1$ (INDEF, INCENSE$_2$, IN α)]
16. [kaffe$_1$bränn$_2$are]$_3$ = LOC$_3^{\alpha}$; [BURN$_2$ (INDEF, COFFEE$_1$, IN α)]
17. [accroche-$_1$casseroles$_2$]$_3$ = LOC$_3^{\alpha}$; [HANG$_1$ (INDEF, SAUCEPAN$_2$, ON α)]
18. [kastrull$_1$häng$_2$are]$_3$ = LOC$_3^{\alpha}$; [HANG$_2$ (INDEF, SAUCEPAN$_1$, ON α)]

4.4 Causative

Some of the rare French VN compounds that accept unaccusative and unergative Vs receive a reading involving a causative relation. We assume a same semantic structure for both cases: an additional argument (the causer or Actor) adds to the V, and the N is interpreted as an Undergoer (not acting entirely volitionally) of the V (cf. [17]). For example, *trotte-bébé* 'toddle-baby=baby walker', is a device that makes the baby toddle. In Swedish, *folkförödare* 'people+devastater=tuberculosis' involves a causative relation.

19. [trotte-$_1$bébé$_2$]$_3$= DEVICE$_3^{\alpha}$; [CAUSE (α (TODDLE$_1$ (BABY$_2$)))]
20. [folk$_1$föröd$_2$are]$_3$ = DISEASE$_3^{\alpha}$; [CAUSE (α (DEVASTATE$_2$ (PEOPLE$_1$)))]

4.5 Undergoer

Exceptionally, a few French VN compounds have an Undergoer interpretation, in which the N constituent, instead, is an Actor. This case is thus the opposite of the Agent case in sub-section 4.1. For example, *croque-monsieur* 'crunch-sir=toast (that

the sir crunches)', or *pique-poule* (normally spelled as *picpoul*) 'pick-hen=grape (picked by hens)'. This meaning is unproductive in contemporary French, and seems to be ruled out for Swedish NV-*are* compounds.

21. [croque-$_1$monsieur$_2$]$_3$= UND$_3^{\alpha}$; [CRUNCH$_1$ (SIR$_2$, α)]

4.6 Place and Event

Apart from the output meanings above, French VN compounds can denote the place, where the event expressed takes place: *coupe-gorge* 'cut-throat=dangerous place where one risks having one's throat cut'. They are often toponyms, such as *Chante-merle* 'sing-blackbird=a place where the blackbirds sing'. We have not attested any Swedish NV-*are* compound with a Place meaning (cf. *diner* in English).

22. [coupe-$_1$gorge$_2$]$_3$= PLACE$_3$; [LOC (CUT$_1$ (INDEF, THROAT$_2$))]
23. [Chante-$_1$merle$_2$]$_3$= PLACE$_3$; [LOC (SING$_1$ (BLACKBIRD$_2$))]

In addition, French VN and Swedish NV-*are* compounds can denote the event itself expressed by the compound, such as *höftrullare* 'hip roller=rolling the hip'. Some of the compounds with an Event meaning can, according to context, have an additional result interpretation, e.g. *baise-main* kiss-hand 'the act of kissing a hand' vs. 'hand-kiss'.

24. [höft$_1$rull$_2$are]$_3$ = EVENT$_3$; ROLL$_1$ (HIP$_2$)
25. [baise-$_1$main$_2$]$_3$= EVENT$_3$; KISS$_1$ (INDEF, HAND$_2$)

The Place and Event cases do not involve any linking variable. The compound's output meaning does not correspond to a participant in the argument structure of the V; the N constituent can either be an Actor of a V, taking one argument, or an Undergoer of a V, taking two arguments.

5 Action Modality

In [18] a distinction is made between event and non-event English *-er* nominals, corresponding more or less to the distinction between stage-level and individual-level nominals [7], [19]. Inheritance of complement and argument structure correlates with the event interpretation, whereas instruments and occupations, which do not presuppose the existence of an event, typically are non-events. Busa [20], instead, claims that all agentive nominals are best characterized in terms of events, and distinguishes between a changeable property for stage-level nominals, encoded as an Agentive role (cf. 26), and a persistent property for individual-level nominals, encoded as a Telic role (cf. 27):

26. passenger

QUALIA = FORMAL = person
AGENTIVE = travel on vehicle

27. smoker

QUALIA = FORMAL = person
TELIC = smoke

Moreover, Busa [20] argues that state predicates of individual-level nominals can also encode for an agentive role, such as Habit for *smoker* or Ability for *violinist*:

28. violinist

QUALIA = FORMAL = person
TELIC = play violin
AGENTIVE = ability to play violin

29. smoker

QUALIA = FORMAL = person
TELIC = smoke
AGENTIVE = habit to smoke

Jackendoff [6], referring to [21], emphasizes that action modality is an important component for the interpretation and lexical representation of agentive nominals, and not only a matter of pragmatics. There are five major types:

- Current (e.g. *gâte-fête, festförstörare*, 'party trasher')
- Ability (e.g. *gobe-mouches, flugsnappare* 'fly catcher')
- Habit (e.g. *rabat-joie, glädjedödare* 'killjoy')
- Occupation (e.g. *croque-mort, likbärare* 'pall bearer')
- Proper function (*ouvre-boîte, burköppnare* 'can opener')

The Current modality refers to a specific activity on a specific occasion. It concerns stage-level nominals and encodes as an Agentive role. Ability presupposes a potential event (may or may not occur), whilst Habit presupposes repetitive events. Occupation regards persons, practicing the profession indicated by the compound. Proper function concerns objects, and is true irrespectively of actual situations. Thus, the last four modalities involve state predicates and encode as a Telic role, but could additionally encode for an Agentive role [20].

In Table 3, we relate the semantic structures within French VN and Swedish NV-*are* compounds to action modalities. We see that only Proper Function is relevant for objects, all other modalities concern Actors. None of the modalities are relevant for compounds with Place or Event meanings. According to our data, Current (stage-level interpretation) is rarely lexicalized among French VN and Swedish NV-*are* compounds.

Table 3. Semantic structures in relation to action modalities

ACTOR (Arg1)	Current	Habit	Ability	Occupation
CAUSATIVE	Current	Habit	Ability	?
INSTRUMENT	Proper function			
LOCATIVE	Proper function			
PLACE	?			
EVENT	?			

In relation to the general notion of modality, action modality is normally labelled as dynamic, which can be abilitive or volitive [22]. Nuyts [23] proposes dynamic modality to be a subcategory of quantificational aspect, since notions such as "ability/potential" and "need" are semantically similar to notions such as "iterative", "habitual" and "generic". Furthermore, action modality is not overtly linguistically coded and does not affect the lexical content of the verb stem [24]. It can be lexicalized and does not depend solely on context for its interpretation. Hence, action modality, which cannot be defined as an attitudinal expression, seems to be a sort of objective modality [25]. In our opinion, we cannot really see the need for this notion: we propose that the Agentive can be left underspecified for action modality, and that only the Telic is important for the lexical representation of agentive nominal compounds in French and Swedish.

6 GL Representations of the Most Frequent Cases

In order for our study to have some predictive power and importance for NLP systems, we focus on the lexical representation of the three most frequent cases, namely Actor (where the compound is the Arg1 of the V), Instrument and Locative, in which the N constituent is an Undergoer and the compound an Actor in relation to the V. Another possible analysis, different from ours, would be to consider the Actor interpretation as a case of lexical underspecification [26]. The other semantic structures and output meanings, some of them unproductive, are marginal and can probably be exhausted. Nevertheless, instead of proposing a single lexical rule, with a common denominator (cf. [17] for French VN compounds), we propose different lexical representations. The output meanings of the compounds are assumed to be specified in the Type structure, and their internal semantic structure in the Telic role. We do not assume an "instrumental subject"-interpretation of compounds with Instrument or Locative meanings: '*a can opener opens can' or '*a clothes hanger hangs clothes' are not well-formed, in our opinion. Instead, we introduce a default argument, preferably a human agent (not a user *w*, cf. [21]). We use a simplified form of the GL [7], [27], and omit for example Constitutive and Agentive in the Qualia structure.

30. *porte-drapeau*, *fanbärare* 'flag bearer'

$$\begin{bmatrix} \text{TYPESTR} = & [\,\text{ARG1} = \text{x: human}\,] \\ \text{ARGSTR} = & [\text{D-ARG1} = \text{y: flag}\,] \\ \text{EVENTSTR} = & [\text{D-E1} = \text{e: process}\,] \\ \text{QUALIA} = & \begin{bmatrix} \text{FORMAL} = \text{x} \\ \text{TELIC} = \text{bear\_flag\_act (e, x, y)} \end{bmatrix} \end{bmatrix}$$

31. *ouvre-boîte*, *burköppnare* 'can opener'

$$\begin{bmatrix} \text{TYPESTR} = & [\,\text{ARG1} = \text{z: artefact\_instrument}\,] \\ \text{ARGSTR} = & \begin{bmatrix} \text{D-ARG1} = \text{x: human} \\ \text{D-ARG2} = \text{y: can} \end{bmatrix} \\ \text{EVENTSTR} = & [\text{D-E1} = \text{e: process}\,] \\ \text{QUALIA} = & \begin{bmatrix} \text{FORMAL} = \text{z} \\ \text{TELIC} = \text{open\_can\_act (e, x, y, with z)} \end{bmatrix} \end{bmatrix}$$

32. *accroche-casseroles, kastrullhängare* 'saucepan hanger'

$$\begin{bmatrix} \text{TYPESTR} = & [\text{ARG1} = \text{z: artefact\_locative}] \\ \text{ARGSTR} = & \begin{bmatrix} \text{D-ARG1} = \text{x: human} \\ \text{D-ARG2} = \text{y: saucepan} \end{bmatrix} \\ \text{EVENTSTR} = & [\text{D-E1} = \text{e: process}] \\ \text{QUALIA} = & \begin{bmatrix} \text{FORMAL} = \text{z} \\ \text{TELIC} = \text{hang\_saucepan\_act (e, x, y, on z)} \end{bmatrix} \end{bmatrix}$$

Note that *x* in the representations (30-32) can be filled with any entity able to display an Actor role. Likewise, *y* can be filled with any entity manifesting an Undergoer role. The events can also be of different types. Our data seems to indicate that intransitive Vs and N constituents with Place or Time meanings (roles displayed by adjuncts in syntax) occur especially in compounds with Actor (Arg1) meanings. In sum, the specification of arguments and predicate structures in the Qualia is important for the analysis of compounds: those included here are all linked to the Telic. Furthermore, phrase structure schemes could be used to account for their compounding (cf. [27]).

7 Discussion

Our analysis of compounds is domain independent, and aims at general semantic structures (cf. [28]), supposed to be lexicalized and more or less productive. Through knowledge about productive semantic patterns, new compounds are created and interpreted [29]. Odd interpretations of compounds are in fact rare [30]. Lapata [31] underlines three problems that compounds still pose for automatic interpretation within NLP: (i) their high productivity implies a need to interpret previously unseen formations; (ii) their internal semantic relation is often implicit; (iii) context and pragmatics have impact on their interpretation.

Contextual information can help to disambiguate unknown compounds of the types included in our study: e.g. subject position in combination with Actor (Arg1) interpretation, "with" in combination with Instruments, and "in" or "on" in combination with Locatives (cf. however [30] for the problematic distinction between agent and instrument at both the morphological and the syntactic level). Since the V constituents in French VN and Swedish NV-*are* compounds cannot always occur as independent Ns in syntax (**porte-*, **häng-/*hängare*), it is not possible to map each of the constituents onto a conceptual representation as is possible for NN root compounds (cf. the systems of [32], [33]). However, a disambiguation algorithm can map the V constituents to their respective verbs and examine distributional properties: e.g. retrieve frequencies of the verb's relation to its objects (verb-argument tuples). In the majority of cases, the N constituent is an internal argument of the (transitive) V constituent. The set of possible interpretations provided by our study enables manual disambiguation of compounds in context, which then can be added to the lexicon.

Our unified account of French VN and Swedish NV-*are* compounds can have relevance for MT or other multi-lingual language processing tasks with regard to Romance and Germanic languages: the GL representation constitutes a neutral platform [27], [32]. Cartoni [34] proposes a prototype of a MT system for handling constructed neologisms, and to which our analysis could be fitted. The first module

checks unknown words with regard to their being potentially constructed or not. If they are, it performs a morphological analysis of their structure and lexeme-bases. The second module generates a possible translation of the analyzed construction. The prototype relies on lexical resources and a set of bilingual Lexeme Formations Rules. The lexeme-bases are checked against the lexical resources and the rules provide information of how to translate them into the target language (e.g. French $V_xN_y \rightarrow$ Swedish $N_yV_x$are: *brise-glace* 'break-ice=icebreaker'$\rightarrow$ *isbrytare*, or alternatively, if the French V constituent corresponds to a lexically established N in Swedish, French $V_xN_y \rightarrow$ Swedish N_yN_x: *appuie-tête* 'rest-head=headrest' $\rightarrow$ *huvudstöd*).

8 Conclusion

This study has attempted to provide a unified account of the complex semantics of French VN and Swedish NV-*are* compounds. We have adopted the frameworks of Jackendoff [6] and GL [7], [27], and been able to find some general semantic structures giving rise to particular output meanings. In the most productive semantic structures, the compounds as well as the N constituents display a role in the argument structure of the V constituent. We assume the Telic role in the Qualia to be most important for their lexical representation. Contrary to the opinion expressed in [20], we suggest that the Agentive role can be left un(der)specified, since it does not add much to their disambiguation or analysis. In conclusion, we hope that our study can have application for NLP systems. Possible applications could be to elaborate a probabilistic algorithm dealing with a disambiguation task for unseen compounds within domain-independent unrestricted text. Our unified account also has relevance for machine translation between French and Swedish, and for multi-lingual language processing with regard to Romance and Germanic languages.

Dictionaries

Fransk-svensk ordbok. (1995). Natur och kultur, Stockholm.
Norstedts fransk-svenska ordbok. (1993). Norstedt, Stockholm.
Norstedts stora fransk-svenska ordbok. (1998). Norstedt, Stockholm.
Norstedts stora svensk-franska ordbok. (1998). Norstedt, Stockholm.
TLFi, Le Trésor de La Langue Française informatisé. http://atilf.atilf.fr/tlf.htm
SAOB, Svenska Akademiens Ordbok. http://g3.spraakdata.gu.se/saob/

References

1. Booij, G.: Compounding and Construction Morphology. In: Lieber, R., Štekauer, P. (eds.) The Oxford Handbook of Compounding, pp. 201-216. Oxford University Press, Oxford (2009)

2. Bisetto, A.: Italian Compounds of the *Accendigas* Type: A Case of Endocentric Formations?. In: Bouillon, P., Estival, D. (eds.) Proceedings of the Workshop on Compound Nouns: Multilingual Aspects of Nominal Composition, pp. 77-87. ISSCO, Geneva (1994)
3. Lieber, R.: Deconstructing Morphology: Word Formation in Syntactic Theory. University of Chicago Press, Chicago (1992)
4. Corbin, D.: Hypothèses sur les frontières de la composition nominale. Cahiers de grammaire 17, 25-55 (1992)
5. Booij, G.: The Grammar of Words. Oxford University Press, Oxford (2005)
6. Jackendoff, R.: Compounding in the Parallel Architecture and Conceptual Semantics. In: Lieber, R., Štekauer, P. (eds.) The Oxford Handbook of Compounding, pp. 105-128. Oxford University Press, Oxford (2009)
7. Pustejovsky, J.: The Generative Lexicon. MIT Press, Cambridge, MA (1995)
8. Van Valin, R. D., Jr.: Semantic Macroroles in Role and Reference Grammar. In: Kailuweit, R., Hummel, M. (eds.) Semantische Rollen, pp. 62-82. Narr, Tübingen (2002)
9. Vendler, Z.: Verbs and Times. The Philosophical Review 66, 143-160 (1957)
10. Burzio, L.: Italian Syntax: A Government-Binding Approach. Reidel, Dordrecht (1986)
11. Perlmutter, D. M.: The Split Morphology Hypothesis. In: Hammond, M., Noonan, M. (eds.) Theoretical Morphology: Approaches in Modern Linguistics, pp. 79-100. Academic Press, San Diego (1988)
12. Fradin, B.: On a Semantically Grounded Difference between Derivations and Compounding. In: Dressler, W. U. *et al.* (eds.) Morphology and its Demarcations, pp. 161-182. Benjamins, Amsterdam/Philadelphia (2005)
13. Dowty, D.: Thematic Proto-Roles and Argument Selection. Language 67, 547-619 (1991)
14. Rosenberg, M.: La formation agentive en français : les composés $[VN/A/Adv/P]_{N/A}$ et les dérivés V-*ant*, V-*eur* et V-*oir*(*e*). PhD Dissertation, Department of French, Italian and Classical Languages, Stockholm University (2008)
15. Dressler, W. U.: Explanation in Natural Morphology, Illustrated with Comparative and Agent-Noun Formation. Linguistics 24, 519-548 (1986)
16. Lieber, R.: Morphology and Lexical Semantics. Cambridge University Press, Cambridge (2004)
17. Roussarie, L.,Villoing, F.: Some Semantic Investigation of the French VN Construction. In: Bouillon, P., Kanzaki, K. (eds.) Proceedings of the Second International Workshop on Generative Approaches to the Lexicon, Geneva, Switzerland (2003).
18. Rappaport Hovav, M., Levin, B.: *-er* Nominals: Implications for the Theory of Argument Structure. In: Stowell, T., Wehrli, E. (eds.) Syntax and the Lexicon. Syntax and Semantics 26, pp. 127-153. Academic Press, San Diego, CA (1992)
19. Carlson, Gregory N. 1977. *Reference to Kinds in English*. PhD Dissertation, University of Massachusetts, Amherst.
20. Busa, F.: The Semantics of Agentive Nominals. In: Saint-Dizier, P. (ed.) Proceedings of ECAI Workshop on Predicative Forms for the Lexicon. Toulouse, France (1996)
21. Busa, F. Compositionality and the Semantics of Nominals. PhD Dissertation, Department of Computer Science, Brandeis University (1996)
22. Palmer, F. R.: Mood and Modality. 2nd ed. Cambridge University Press, Cambridge (2001)
23. Nuyts, J.: The Modal Confusion. In Klinge, A., Müller, H. H. (eds.) Modality: Studies in Form and Function, pp. 5-38. Equinox, London (2005)
24. Bybee, J. L.: Morphology: A Study of the Relation between Meaning and Form. Benjamins, Amsterdam/Philadelphia (1985)
25. Herslund, M.: Subjective and Objective Modality. In: Klinge, A., Müller, H. H. (eds.), Modality: Studies in Form and Function, pp. 39-48. Equinox, London (2005)
26. Pustejovsky, J.: The Semantics of Lexical Underspecification. Folia Linguistica 32, 323-348 (1998)

27. Johnston, M., Busa, F.: Qualia Structure and the Compositional Interpretation of Compounds. In: Viegas, E. (ed.) Breadth and Depth of Semantic Lexicons, pp. 167-189. Kluwer, Dordrecht (1999)
28. Fabre, C.: Interpretation of Nominal Compounds: Combining Domain-Independent and Domain-Specific Information. In: Proceedings of the 16th Conference of Computational Linguistics, pp. 364-369. Copenhagen, Denmark (1996)
29. Ryder, M. E.: Ordered Chaos: The Interpretation of English Noun-Noun Compounds. University of California Press, Berkeley (1994)
30. Isabelle, P.: Another Look at Nominal Compounds. In: Proceedings of the 10th International Conference on Computational Linguistics and the 22nd Annual Meeting of the Association for Computational Linguistics, pp. 509-516. Stanford, California (1984)
31. Lapata, M.: The Disambiguation of Nominalizations. Computational Linguistics 28, 357-38 (2002).
32. McDonald, D. B.: Understanding Noun Compounds. PhD Dissertation, Carnegie Mellon University, Pittsburgh, Pennsylvania (1982)
33. Finin, T.: The Semantic Interpretation of Nominal Compounds. In: Proceedings of First Annual National Conference on Artificial Intelligence, pp. 310-315. Stanford, California (1980)
34. Cartoni, B.: Lexical Morphology in Machine Translation: A Feasibility Study. In: Proceedings of the 12th Conference of the European Chapter of the ACL, pp. 130-38. Association for Computational Linguistics (2009).

Computing Linear Discriminants for Idiomatic Sentence Detection

Jing Peng[1], Anna Feldman[1,2], and Laura Street[2]

[1] Department of Computer Science
[2] Department of Linguistics
Montclair State University
Montclair, NJ 07043, USA
{pengj,feldmana,streetl1}@mail.montclair.edu

Abstract. In this paper, we describe the binary classification of sentences into idiomatic and non-idiomatic. Our idiom detection algorithm is based on linear discriminant analysis (LDA). To obtain a discriminant subspace, we train our model on a small number of randomly selected idiomatic and non-idiomatic sentences. We then project both the training and the test data on the chosen subspace and use the three nearest neighbor (3NN) classifier to obtain accuracy. The proposed approach is more general than the previous algorithms for idiom detection — neither does it rely on target idiom types, lexicons, or large manually annotated corpora, nor does it limit the search space by a particular linguistic construction.

1 Introduction

Previous work on automatic idiom classification has typically been of two types: those which make use of type-based classification methods (Lin, 1999; Baldwin et al., 2002; Fazly and Stevenson, 2006; Bannard, 2007; Fazly et al., 2009) and those which make use of token-based classification methods (Birke and Sarkar, 2006; Katz and Giesbrecht, 2006; Fazly et al., 2009; Sporleder and Li, 2009). Type-based classification methods recognize idiomatic expressions (=types) to include in a lexicon and typically rely on the notion that many idioms share unique properties with one another. For instance, several idioms are composed of verb-noun constructions (e.g., *break a leg*, *get a grip*, *kick the bucket*) that cannot be altered syntactically or lexically (e.g., *break a skinny leg*, *a grip was got*, *kick the pail*). These unique properties are used to distinguish idiomatic expressions from other types of expressions in a text. Token-based classification methods recognize a particular usage (literal vs. non-literal) of a potentially idiomatic expression. Both of these approaches view idioms as multi-word expressions (MWEs) and rely crucially on preexisting lexicons or manually annotated data. They also tend to limit the search space by a particular type of linguistic construction (e.g., Verb+Noun combinations). The task of automatic idiom classification is extremely important for a variety of NLP applications; e.g., a machine translation system must translate *held fire* differently in *The army held their fire* and *The worshippers held the fire up to the idol* (Fazly et al., 2009).

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 17-28

Received 23/11/09
Accepted 16/01/10
Final version 10/03/10

2 Our Approach

Unlike previous work on idiom detection, we view the solution to this problem as a two-step process: 1) filtering out sentences containing idioms; 2) extracting idioms from these filtered out sentences. In our current work we only consider step 1, and we frame this task as one of classification. We believe that the result of filtering out idiomatic sentences is already useful for many applications such as machine translation, information retrieval, or foreign/ second language instruction, e.g., for effective demonstrations of contexts in which specific idioms might occur

Our idiom detection algorithm is based on linear discriminant analysis (LDA). To obtain a discriminant subspace, we train our model on a small number of randomly selected idiomatic and non-idiomatic sentences. We then project both the training and the test data on the chosen subspace and use the three nearest neighbor (3NN) classifier to obtain accuracy. The proposed approach is more general than the previous algorithms for idiom detection — neither does it rely on target idiom types, lexicons, or large manually annotated corpora, nor does it limit the search space by a particular type of linguistic construction. The following sections describe the algorithm, the data and the experiments in more detail.

2.1 Idiom Detection based on Discriminant Analysis

The approach we are taking for idiomatic sentence detection is based on linear discriminant analysis (LDA) (Fukunaga, 1990). LDA often significantly simplifies tasks such as regression and classification by computing low-dimensional subspaces having statistically uncorrelated or discriminant variables. In language analysis, statistically uncorrelated or discriminant variables are extracted and utilized for description, detection, and classification. Woods et al. (1986), for example, use statistically uncorrelated variables for language test scores. A group of subjects was scored on a battery of language tests, where the subtests measured different abilities such as vocabulary, grammar or reading comprehension. Horvath (1985) analyzes speech samples of Sydney speakers to determine the relative occurrence of five different variants of each of five vowels sounds. Using this data, the speakers clustered according to such factors as gender, age, ethnicity and socio-economic class.

LDA is a class of methods used in machine learning to find the linear combination of features that best separate two classes of events. LDA is closely related to principal component analysis (PCA), where a linear combination of features that best explains the data. Discriminant analysis explicitly exploits class information in the data, while PCA does not.

Idiom detection based on discriminant analysis has several advantages. First, it does not make any assumption regarding data distributions. Many statistical detection methods assume a Gaussian distribution of normal data, which is far from reality. Second, by using a few discriminants to describe data, discriminant

analysis provides a compact representation of the data, resulting in increased computational efficiency and real time performance.

2.2 Linear Discriminant Analysis

In LDA, within-class, between-class, and mixture scatter matrices are used to formulate the criteria of class separability. Consider a J class problem, where m_0 is the mean vector of all data, and m_j is the mean vector of jth class data. A within-class scatter matrix characterizes the scatter of samples around their respective class mean vector, and it is expressed by

$$S_w = \sum_{j=1}^{J} p_j \sum_{i=1}^{l_j} (x_i^j - m_j)(x_i^j - m_j)^t, \tag{1}$$

where l_j is the size of the data in the jth class, p_j ($\sum_j p_j = 1$) represents the proportion of the jth class contribution, and t denotes the transpose operator. A between-class scatter matrix characterizes the scatter of the class means around the mixture mean m_0. It is expressed by

$$S_b = \sum_{j=1}^{J} p_j (m_j - m_0)(m_j - m_0)^T. \tag{2}$$

The mixture scatter matrix is the covariance matrix of all samples, regardless of their class assignment, and it is given by

$$S_m = \sum_{i=1}^{l} (x_i - m_0)(x_i - m_0)^T = S_w + S_b. \tag{3}$$

The Fisher criterion is used to find a projection matrix $W \in \Re^{q \times d}$ that maximizes

$$J(W) = \frac{|W^t S_b W|}{|W^t S_w W|}. \tag{4}$$

In order to determine the matrix W that maximizes $J(W)$, one can solve the generalized eigenvalue problem: $S_b w_i = \lambda_i S_w w_i$. The eigenvectors corresponding to the largest eigenvalues form the columns of W. For a two class problem, it can be written in a simpler form: $S_w w = m = m_1 - m_2$, where m_1 and m_2 are the means of the two classes. In practice, the small sample size problem is often encountered, when $l < q$. In this case S_w is singular. Therefore, the maximization problem can be difficult to solve.

2.3 Margin Criterion for Linear Dimensionality Reduction

For idiomatic sentence detection, we propose an alternative to the Fisher criterion. Here we first focus on two class problems. We note that the goal of LDA is

to find a direction w that simultaneously places two classes afar and minimizes within class variations. Fisher's criterion 4 achieves this goal. Alternatively, we can achieve this goal by maximizing

$$J(w) = tr(w^t(S_b - S_w)w), \tag{5}$$

where tr denotes the trace operator. Notice that $tr(S_b)$ measures the overall scatter of class means. Therefore, a large $tr(S_b)$ implies that the class means spread out in a transformed space. On the other hand, a small $tr(S_w)$ indicates that in the transformed space the spread of each class is small. Thus, when maximized, J indicates that data points are close to each other within a class, while they are far from each other if they come from different classes.

To see that our proposal (Eq. 5) is margin based, notice that maximizing $tr(S_b - S_w)$ is equivalent to maximizing $J = \frac{1}{2}\sum_i^2 \sum_j^2 p_i p_j d(C_i, C_j)$, where p_i denotes the probability of class C_i. The interclass distance d is defined as $d(C_i, C_j) = d(m_i, m_j) - tr(S_i) - tr(S_j)$, where m_i represents the mean of class C_i, and S_i represents the scatter matrix of class C_i. Here $d(C_i, C_j)$ measures the average margin between two classes. Therefore, maximizing our objective produces large margin linear discriminants. Large margin discriminants often result in better generalization (Vapnik, 1998). In addition, there is no need to calculate the inverse of S_w, thereby avoiding the small sample size problem associated with the Fisher criterion.

3 Computing Linear Discriminants with Semi-Definite Programming

Suppose that w optimizes (5). So does cw for any constant $c \neq 0$. Thus we require that w have unit length. The optimization problem then becomes

$$\begin{aligned} &\max_w \; tr(w^t(S_b - S_w)w) \\ &\text{subject to: } \|w\| = 1. \end{aligned}$$

This is a constraint optimization problem. Since $tr(w^t(S_b - S_w)w) = tr((S_b - S_w)ww^t) = tr((S_b - S_w)X)$, where $X = ww^t$, we can rewrite the above constraint optimization problem as

$$\begin{aligned} \max_X \; & tr((S_b - S_w)X) \\ & I \bullet X = 1 \\ & X \succeq 0 \end{aligned} \tag{6}$$

where I is the identity matrix and the inner product of symmetric matrices is $A \bullet B = \sum_{i,j}^n a_{ij} b_{ij}$, and $X \succeq 0$ means that the symmetric matrix X is positive semi-definite. Indeed, if X is a solution to the above optimization problem, then $X \succeq 0$ and $I \bullet X = 1$ implies $\|w\| = 1$, assuming $\text{rank}(X) = 1$.

The above problem is a semi-definite program (SDP), where the objective is linear with linear matrix inequality and affine equality constraints. Because linear matrix inequality constraints are convex, SDPs are convex optimization problems. The significance of SDP is due to several factors. SDP is an elegant generalization of linear programming, and inherits its duality theory. For a comprehensive overview on SDP, see (Vandenberghe and Boyd, 1996).

SDPs arise in many applications, including sparse PCA, learning kernel matrices, Euclidean embedding, and others. In general, generic methods are rarely used for solving SPDs, because their time grows at the rate of $O(n^3)$ and their memory grows in $O(n^2)$, where n is the number of rows (or columns) of a semidefinite matrix. When n is greater than a few thousands, SDPs are typically not used. However, there are algorithms that have a good theoretical foundation to solve SDPs (Vandenberghe and Boyd, 1996). In addition, semidefinite programming is a very useful technique for solving many problems. For example, SDP relaxations can be applied to clustering problems such that after solving a SDP, final clusters can be computed by projecting the data onto the space spanned by the first few eigenvectors of the SDP solution. For large-scale problems, there is a tremendous opportunity for exploiting special structures in problems, as those suggested in (Ben-Tal and Nemirovski, 2004; Nesterov, 2003).

Assume $\text{rank}(X) = 1$. Since X is symmetric, one can show that $\text{rank}(X) = 1$ iff $X = ww^t$ for some vector w. Therefore, we can recover w from X as follows. Select any column (say the ith column) of X such that $X(1,i) \neq 0$, and let

$$w = X(:,i)/X(1,i), \tag{7}$$

where $X(:,i)$ denotes the ith column of the matrix X. Thus, our goal here is to ensure the solution X to the above constraint optimization problem has rank at most 1.

One way to guarantee $\text{rank}(X) = 1$ is to use $\text{rank}(X) = 1$ as an additional constraint in the optimization problem. However, the constraint $\text{rank}(X) = 1$ is not convex and the resulting problem is difficult to solve. It turns out that the above formulation (6) is sufficient to ensure that the rank of the optimal solution X to Eq. (6) is one, i.e., $\text{rank}(X) = 1$.

Theorem 1. *Let X be the solution to the semi-definite program (6). Also, let $rank(X) = r$. Then $r = rank(X) = 1$.*

The proof of the theorem is in Appendix A. The theorem states that our procedure for computing w from the matrix X (Eq. 7) is guaranteed to produce the correct answer. We call our algorithm SDP-LDA. An attractive property associated with our algorithm is that it does not have any procedural parameters. Thus, it does not require expensive cross-validation to determine its optimal performance.

4 Dataset

In our experiments, we used the dataset described by Fazly et al. (2009). This is a dataset of verb-noun combinations extracted from the British National Cor-

pus (BNC, Burnard (2000)). The VNC tokens are annotated as either literal, idiomatic, or unknown. The list contains only those VNCs whose frequency in BNC was greater than 20 and that occurred at least in one of two idiom dictionaries (Cowie et al., 1983; Seaton and Macaulay, 2002). The dataset consists of 2,984 VNC tokens[3].

Since our task is framed as sentence classification rather than MWE extraction and filtering, we had to translate this data into our format. Basically, our dataset has to contain sentences with the following tags: I (=idiomatic sentence), L (=literal), and Q (=unknown). Translating the VNC data into our format is not trivial. A sentence that contains a VNC idiomatic construction can be unquestionably marked as I (=idiomatic); however, a sentence that contains a non-idiomatic occurrence of VNC cannot be marked as L since these sentences could have contained other types of idiomatic expressions (e.g., prepositional phrases) or even other figures of speech. So, by marking automatically all sentences that contain non-idiomatic usages of VNCs, we create an extremely noisy dataset of literal sentences. The dataset consists of 2,550 sentences, of which 2,013 are idiomatic sentences and the remaining 537 are literal sentences.

5 Experiments

We first apply the bag-of-words model to create a term-by-sentence representation of the 2,550 sentences in a 6,844 dimensional term space. The Google stop list is used to remove stop words.

We randomly choose 300 literal sentences and 300 idiomatic sentences as training and randomly choose 100 literals and 100 idioms from the remaining sentences as testing. Thus the training dataset consists of 600 examples, while the test dataset consists of 200 examples. We train our model on the training data and obtain one discriminant subspace. We then project both training and test data on the chosen subspace. Note that for the two class case (literal vs. idiom), one dimensional subspace is sufficient. In the reduced subspace, we compare three classifiers: the three nearest neighbor (3NN) classifier, the quadratics classifier that fits multivariate normal densities with covariance estimates stratified by classes (Krzanowski, 1988), and support vector machines (SVMs) with the Gaussian kernel (Cristianini and Shawe-Taylor, 2000). The kernel parameter was chosen through 10 fold cross-validation. We repeat the experiment 10 times to obtain the average accuracy rates registered by the three methods. The following table shows the accuracy rates over the ten runs.

We compare the proposed technique against a random baseline approach. The baseline approach flips a fair coin. If the outcome is head, it classifies a given sentence as idiomatic. If the outcome is tail, it classifies a given sentence as a regular sentence.

Even though we used Fazly et al. (2009)'s dataset for these experiments (see Section 4), the direct comparison with their methods is impossible here because

[3] To read more about this dataset, the reader is referred to Cook et al. (2008)

3NN	Quadratic	SVMs	Baseline
0.8015	0.7690	0.7890	0.50

Table 1. Classification accuracy rates computed by the three competing methods compared against the baseline.

our tasks are formulated differently. Fazly et al. (2009)'s unsupervised model that relies on the so-called canonical forms (CForm) gives 72.4% (macro-)accuracy on the extraction of idiomatic tokens when evaluated on their test data.

6 Analysis

To gain insights into the performance of the proposed technique, we created a dataset that is manually annotated to avoid noise in the literal dataset. We asked three human subjects to annotate 200 sentences from the VNC dataset as idiomatic, non-idiomatic or unknown. 100 of these sentences contained idiomatic expressions from the VNC data. We then merged the result of the annotation by the majority vote.

We also measured the inter-annotator agreement (the Cohen kappa k, Cohen (1960); Carletta (1996)) on the task. Interestingly, the Cohen kappa coefficient was much higher for the idiomatic data than for the so-called literal data: k (idioms) = 0.91; k (literal) = 0.66. There are several explanations of this performance. First, the idiomatic data is much more homogeneous since we selected sentences that already contained VNC idiomatic expressions. The rest of the sentences might have contained metaphors or other figures of speech and thus the judgments were more difficult to do. Second, humans easily identify idioms, but the decision whether a sentence is literal or figurative is much more challenging. The notion of "figurativeness" is not a binary property (as might be suggested by the labels that were available to the annotators). "Figurativeness" falls on a continuum from completely transparent (= literal) to entirely opaque (=figurative)[4] Third, the human annotators had to select the label, literal or idiomatic, without having access to a larger, extra-sentential context, which might have affected their judgements. Although the boundary between idiomatic and literal expressions is not entirely clear (expressions do seem to fall on a continuum in terms of idiomaticity), some expressions are clearly idiomatic and others clearly literal based on the overall agreement of our annotators. By classifying sentences as either idiomatic or literal, we believe that this additional sentential context could be used to further investigate how speakers go about making these distinctions.

[4] A similar observation is made by Cook et al. (2008) with respect to idioms.

7 Discussion

Below we provide output sentences identified by our algorithm as either idiomatic or literal.

1. True Positives (TP): Idiomatic sentences identified as idiomatic
   - *We lose our temper, feel cornered and frightened, it can be the work of an instant.*
   - *Omanis made their mark in history as early as the third century.*
2. False Positives (FP): Non-idiomatic sentences identified as idiomatic
   - *We had words of the sixties, there were words of the seventies, there were words of the eighties, words of the nineties, and we're influencing by those words, actually that's reasonably in popularity and er increasing usage, and sometime we, people actually use it and they don't know what it means.*
   - *Therefore, taking the square root of this measure we get the correlation coefficient.*
3. True Negatives (TN): Non-idiomatic sentences identified as non-idiomatic
   - *It holds up to three horses and will be driven to and from London by Mrs. Charley from their home just outside Coventry.*
   - *The referee blew a toy trumpet and Harry Payne gave the golf club a mighty hit with his bat, breaking the shaft in two.*
4. False Negatives (FN): Idiomatic sentences identified as non-idiomatic
   - *It therefore has a long-term future.*
   - *It has also been agreed that Italy will pay a reciprocal visit to Dublin in April when they will take part in a Four Nations competition to replace the Home.*

Our error analysis reveals that many cases are fuzzy and clear literal/idiomatic demarcation is difficult.

In examining our false positives (i.e., non-idiomatic expressions that were marked as idiomatic by the model), it becomes apparent that the classification of cases is not clear-cut. The expression *words of the sixties/seventies/eighties/nineties* is not idiomatic; however, it is not entirely literal either. It is metonymic – these decades could not literally produce words. Another false positive contains the expression *take the square root.* While seemingly similar to the idiom take root in *plans for the new park began to take root*, the expression *take the square root* is not idiomatic. It does not mean "to take hold like roots in soil." Like the previous false positive, we believe *take the square root* is figurative to some extent. A person cannot literally take the square root of a number like he can literally take milk out of the fridge.

When it comes to classifying expressions as idiomatic or literal, our false negatives (i.e., idiomatic expressions that were marked as non-idiomatic by the model) reveal that human judgments can be misleading. For example, *It therefore has a long-term future* was marked as idiomatic in the test corpus. While our human annotators may have thought that an object could not literally have (or hold) a long-term future, this expression does not appear to be truly idiomatic. We do not consider it to be as figurative as a true positive like *lose our temper.*

Another false negative contains a case of metonymy *Italy will pay a reciprocal visit* and the verbal phrase *take part*. In this case, our model correctly predicted that the expression is non-idiomatic. Properties of metonymy are different from those of idioms, and the verbal phrase *take part* has a meaning separate from that of the idiomatic expression *take someone's part*.

Another interesting feature that we discovered in analyzing our false negatives is that some idiomatic expressions still retain their original meanings even when other words intervene and the idioms' component words are separated and reordered. For example, in the sentence *I was little better than a criminal on whom they must keep tabs*, the prepositional phrase *on whom* is removed from the end of the idiom *keep tabs on whom* and placed in an earlier position in the sentence. Despite this permutation, the idiom still maintains its idiomatic meaning.

All of these observations support Gibbs (1984)'s claim (based on experimental evidence) that the distinctions between literal and figurative meanings have little psychological validity. He views literal and figurative expressions as end points of a single continuum. This makes the task of idiom detection even more challenging because often, perhaps, there is no objective clear boundary between idioms and literal expressions.

8 Conclusion

In this study we did not want to restrict ourselves to idioms of a particular syntactic form. We applied this method to English and used the VNC (Fazly et al., 2009) corpus for our experiments. However, in principle, the technique is language- and structure-independent.

Our binary classification approach has multiple practical applications. It is useful for indexing purposes in information retrieval (IR) as well as for increasing the precision of IR systems. Knowledge of which sentences should be interpreted literally and which figuratively can also improve text summarization and machine translation systems. Applications such as style evaluation or textual steganography detection can directly benefit from the method proposed in this paper as well. Classified sentences are useful for language instruction as well, e.g., for effective demonstrations of contexts in which specific idioms might occur. We also feel that identifying idioms at the sentence level may provide new insights into the kinds of contexts that idioms are situated in. These findings could further highlight properties that are unique to specific idioms if not idioms in general.

Our current work is concerned with improving the detection rates of our model. At present, our model does not use text coherence as a feature, and we think we could significantly improve our performance if we considered a larger context. Once the detection rates of our model have been improved, we will extract idioms from the sentences our model has classified as idiomatic. We have yet to see which method will work the best for this task.

A Proof of Theorem 1

Proof. We rewrite $S_b - S_w = 2S_b - S_m$, where $S_m = S_b + S_w$. Let $null(A)$ denote the null space of matrix A. Since $null(S_m) \subseteq null(S_b)$, there exists a matrix $P \in \Re^{q \times s}$ that simultaneously diagonalizes S_b and S_m Fukunaga (1990), where $s \leq \min\{l-1, q\}$ is the rank of S_m.

The matrix P is given by

$$P = Q\Lambda_m^{-1/2}U,$$

where Λ_m and Q are the eigenvalue and eigenvector matrices of S_m, and U is the eigenvector matrix of $\Lambda_m^{-1/2}Q^tS_bQ\Lambda_m^{-1/2}$. Thus, the columns of P are the eigenvectors of $2\lambda S_b - S_m$ and the corresponding eigenvalues are $2\Lambda_b - I$. We then have

$$P^tS_bP = \Lambda_b, \qquad P^tS_mP = I. \tag{8}$$

where $\Lambda_b = \text{diag}\{\sigma_1, \cdots, \sigma_s\}$.

Consider the range of P over $Y \in \Re^{s \times q}$ with $rank(Y) = s$. The range $W = PY$ includes all $q \times q$ matrices with $rank = s$. Then

$$\begin{aligned}\max_W tr(W^t(2S_b - S_m)W) &= \max_Y tr((PY)^t(2S_b - S_m)PY) \\ &= \max_Y tr(Y^t(2\Lambda_b - I)Y).\end{aligned}$$

It is straightforward to show that the maximum is attained by $Y = [e_1e_2 \cdots e_r; 0]$, where e_i is a vector whose ith component is one and the rest is 0. From this it is clear that $W = PY$ consists of the first r columns of P, i.e., the eigenvectors corresponding to $2\lambda\sigma_i - 1 > 0$.

Now, since $X = WW^t$, we have $X = \sum_{i=1}^r w_iw_i^t$. Thus,

$$tr(X) = \sum_{i=1}^r w_i^tw_i = r.$$

However, the constraint $I \cdot X = 1$ states that $tr(X) = 1$. It follows that $r = 1$. That is, $rank(X) = 1$.

References

1. Baldwin, T., C. Bannard, T. Tanaka, and D. Widdows (2002). An empirical model of multiword expression decomposability. In Proceedings of the ACL 03 Workshop on Multiword expressions: analysis, acquisition and treatment, pp. 89–96.
2. Bannard, C. (2007). A measure of syntactic flexibility for automatically identifying multiword expressions in corpora. In Proceedings of the ACL 07 Workshop on A Broader Perspective on Multiword Expressions, pp. 1–8.
3. Ben-Tal, A. and A. Nemirovski (2004). Non-euclidean restricted memory level method for large-scale convex optimization.
4. Birke, J. and A. Sarkar (2006). A clustering approach to the nearly unsuper- vised recognition of nonliteral language. In Proceedings of the 11th Confer- ence of the European Chapter of the Association for Computational Linguistics (EACL'06), Trento, Italy, pp. 329–226.
5. Burnard, L. (2000). The British National Corpus Users Reference Guide. Oxford University Computing Services.
6. Carletta, J. (1996). Assessing Agreement on Classification Tasks: The Kappa Statistic. Computational Linguistics 22(2), 249–254.
7. Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. Education and Psychological Measurement (20), 37–46.
8. Cook, P., A. Fazly, and S. Stevenson (2008, June). The VNC-Tokens Dataset. In Proceedings of the LREC Workshop: Towards a Shared Task for Multiword Expressions (MWE 2008), Marrakech, Morocco.
9. Cowie, A. P., R. Mackin, and I. R. McCaig (1983). Oxford Dictionary of Current Idiomatic English, Volume 2. Oxford University Press.
10. Cristianini, N. and J. Shawe-Taylor (2000). An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge, UK: Cam- bridge University Press.
11. Fazly, A., P. Cook, and S. Stevenson (2009). Unsupervised Type and Token Identification of Idiomatic Expressions. Computational Linguistics 35 (1), 61–103.
12. Fazly, A. and S. Stevenson (2006). Automatically Constructing a Lexicon of Verb Phrase Idiomatic Combinations. In Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2006), Trento, Italy, pp. 337–344.
13. Fukunaga, K. (1990). Introduction to statistical pattern recognition. Academic Press.
14. Gibbs, R. W. (1984). Literal Meaning and Psychological Theory. Cognitive Science 8, 275–304.
15. Horvath, B. M. (1985). Variation in Australian English. Cambridge: Cambridge University PRess.
16. Katz, G. and E. Giesbrecht (2006). Automatic identification of non- compositional multi-word expressions using latent semantic analysis. In Proceedings of the ACL'06 Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties, Sydney, Australia, pp. 12–19.
17. Krzanowski, W. (1988). Principles of Multivariate Analysis. UK: Oxford

University Press.
18. Lin, D. (1999). Automatic Identification of Non-compositional Phrases. In Proceedings of ACL, College Park, Maryland, pp. 317–324.
19. Nesterov, I. (2003). Smooth minimization of non-smooth functions.
20. Seaton, M. and A. Macaulay (Eds.) (2002). Collins COBUILD Idioms Dictionary(second ed.). HarperCollins Publishers.
21. Sporleder, C. and L. Li (2009). Lexical Encoding of MWEs. In Proceedings of EACL 2009.
22. Vandenberghe, L. and S. Boyd (1996). Semidefinite programming. SIAM Review 38(1), 49–95.
23. Vapnik, V. (1998). Statistical Learning Theory. New York: Wiley.
24. Woods, A., P. Fletcher, and A. Hughes (1986). Statistics in Language Studies. Cambridge: Cambridge University Press.

Robust Temporal Processing: from Model to System

Tommaso Caselli and Irina Prodanof

ILC-CNR, Pisa
firstName.secondName@ilc.cnr.it

Abstract. This paper shows the functioning and the general architecture of an empirically-based model for robust temporal processing of text/discourse. The starting point for this work has been the understanding of how humans process and recognize temporal relations. The empirical results show that the different salience of the linguistic and commonsense knowledge sources of information calls for specific computational components and procedures to deal with them.

1 Introduction

Temporal processing of text/discourse has recently become one of the most active areas in NLP, boosted by the presence of specific markup languages (ISO-TimeML, SemAF/Time Project) and by a growing number of initiatives (CLEF, TERN, SemEval-TempEval2).

Natural languages have a variety of devices to communicate information about events and their temporal organization and the identification of the temporal relations in a text/discourse is not a trivial task. Previous research has explored and analyzed what sources of information are at play when inferring the temporal orders of eventualities such as tense, temporal adverbs, signals, viewpoint aspect, lexical aspect, discourse relations, commonsense and pragmatic knowledge. Most sources of information for inferring a temporal relation very rarely code in an explict and clear-cut way the specific temporal relation holding between two entities (i.e. eventuality - eventuality, eventuality - temporal expression) and this may lead to biases and incorrect tagging. Substantial linguistic processing is required for a system to perform temporal inferences and commonsense knowledge can hardly be encoded in domain independent programs. One of the main issues which has not been answered so far is how the linguistic devices which languages have at disposal to codify temporal relations

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 29-40

Received 23/11/09
Accepted 16/01/10
Final version 09/03/10

interact both with each other and under which conditions they are autonomous, i.e. able to codify a temporal relations between eventualities without the support of non-purely linguistic elements, like discourse structure or world-knowledge based inferences. This calls for the development of procedures and techniques which maximize the role of the sources of information and the conditions under which they are necessary and sufficient to determine the current temporal relation.

This paper presents a general empirically-based model for robust temporal processing of text/discourse. Though the experiments have been conducted on Italian, the model is language independent. The lack of a complete system is mainly due the absence of temporally annotated resources for Italian over which systems could be developed and evaluated. The remaining of the paper is organized as follows: in sect. 2 we illustrate the methodology and the experimental results on the basis of which the model has been developed. Section 3 reports the overall architecture of the model and the functioning of its core components. Finally, sect. 4 presents the conclusion and observations for future work.

2 Linguistic Information and Pragmatic Mechanisms: Defining an Order of Application

Recent psychological studies ([1], [2]) have established correspondences between the formal aspect of the temporal structure of discourse and the mental representations interpreters built. The order in which eventualities are presented in a text/discourse *vs.* their real chronological order, the use of particular tenses, the presence of elements which explicitly mark a temporal relations are all features used when constructing a mental model of a text/discourse. As [3] pointed out, these features are of the kind which can be constructed automatically by information extraction systems. Knowing how these features interact with respect to their different nature, i.e. linguistic *vs.* world-knowledge based, is a necessary step to have robust automatic extraction systems.

To develop a model for temporal processing we have decided to in-

vestigate through an experimental study if it is possible to determine a hierarchical order of application of the linguistic and non-linguistic sources of information and under which conditions purely linguistic information is necessary and sufficient to determine the temporal relations between the entities in analysis. The aim of this study is that of identifying how deep must the computation of information go, that is how many modules must be activated in order to obtain a reliable temporal representation of the text/discourse.

2.1 Methodology

In order to verify the existence of a salience order of the sources of information and to obtain cues on the way the model should be implemented we have elaborated a test which was submitted to two groups of subjects: a first group of 29 subjects, none of them having knowledge in linguistics (Group 1), and a second group of 6 subjects, all MA students in Linguistics. The two groups were submitted with comparable, though not identical, test data, provided their different backgrounds and the level of metalinguistic analysis required.

In both experiments the subjects were presented with a set of 52 discourse excerpts, automatically extracted from the Italian Syntactic Semantic Treebank (ISST), and were asked to temporally order two highlighted eventualities in the discourse segments. To improve the reliability and avoid inconsistency, the subjects were asked to choose the temporal relations among a restricted set of 5 predetermined values, namely BEFORE, AFTER, SIMULTANEOUS, OVERLAP, and NO TEMPORAL RELATION. No binary interpretation of the temporal relations was allowed. In order to discover the existence of a salience order of application of the sources of information and to determine in a reliable way under which conditions linguistic (grammatical and lexical) information is autonomous (i.e. necessary and sufficient) for the identification of the temporal relations with respect to non-purely linguistic i.e. con-textual one, the subjects were asked to state what source of information had *mostly* helped them in the identification of the temporal relation. Similarly to the first task and to keep the experiments under control, we provided the subjects with a predetermined set of possible answers according to their background. Group 1 had at disposal the val-

ues TENSE, TEMPORAL EXPRESSIONS, and NOT SPECIFIED, while Group 2 had a larger set, i.e. TENSE, TEMPORAL EXPRESSIONS, SIGNAL, ASPECT, SEMANTICS and NOT SPECIFIED.

2.2 Data Analysis and Results

In Table 1 we report the results obatined for the identification of temporal relations. The agreement among the subjects have been computed by means of the K statistics.

Table 1. Agreement of the subjects on temporal relation identification

Agreement of temporal relations	K value
Overall agreement	0.58
Agreement in presence of temporal expressions	0.64
Agreement in presence of signals	0.73
Agreement in presence of shifts in tense	0.70

As the results illustrate, the identification of temporal relations is a challenging task. Only in presence of specific *markers* of temporal relations, such as shifts in tense, temporal expressions and signals, the agreement raises up to reliable values. The data have also shown that the temporal representations humans construct are varied: in absence of specific information they are mainly coarse grained, while unique and clear-cut values can be obtained only in presence of explicit information, i.e. of markers which guides the interpretation process.

One of the main results is the identification of a set of *constraints* and *preferences*. The constraints apply both to the role and to the relationships among the sources of information, while the preferences deal with tense patterns and associated temporal relations. Different constraints are activated for each source of information. The constraints can be conceived as representing the condition under which each source of information is a necessary and sufficient element for recovering the correct temporal relation. In particular, we claim that:

- **Constraint 1 (Tense)**: sequences of adjacent eventualities must have different tense meaning, otherwise other sources of information are responsible for their ordering;

- **Constraint 2 (Temporal Expressions)**: temporal expressions may represent explicit information for the ordering of eventualities provided that: a.) they are related to the moment of the event, E, or to a secondary deictic moment, R; b.) if more than one temporal expressions is present, they must stand in an anchoring relation one with each other to signal a temporal relation; c.) in case there is just one temporal expression related to an eventuality, its contribution is relevant for the anchoring of eventuality on the time line but it is null for the determing the temporal relation between two eventualities;
- **Constraint 3 (Signals)**: signals represent salient information for temporal relations only when their semantics is explicit, like for *dopo* [after], *intanto* [simultaneously]. When they are implicit, like *quando* [when], *per* [for/since], they offer ancillary information which reinforces the contribution of other sources, like tense, viewpoint aspect, lexical aspect, temporal expression, and commonsense knowledge;
- **Constraint 4 (Viewpoint and Lexical Aspect)**: these two types of information have a constraint similar to that for tense: when adjacent eventualities have the same values, either for viewpoint or lexical aspect, their knowledge is necessary but not sufficient to determine the temporal relation.

When all other sources of information fail to provide distinguishing cues, commonsense knowledge is used. We claim that commonsense knowledge is the most salient source of information for recovering temporal relations but also the less affordable, since it may introduce biases and disagreement. The identification that a sequence of sentences forms a text/discourse is the pre-condition for the existence of any kind of relations between the discourse entities. It is only in this sense that temporal relations are a by-product of the computation of the general discourse structure, and, as the data have shown, discourse structure cannot be considered the primary source of information for the identification of temporal relations. Of course, knowledge of discourse structure can improve the automatic recognition of temporal relations, as [4] have demonstrated. To illustrate how the constraints work, consider this example:

(1) Marco *è caduto*$_{e1}$. Giovanni l'*ha spinto*$_{e2}$.
Marco fell$_{e1}$. *Giovanni pushed him*$_{e2}$.

In 1 the two eventualities cannot be ordered by exploiting only linguistic information. They are not compliant neither with **Constraint 1** (different tense meaning) nor with **Constraint 4** (different viewpoint and lexical aspect). **Constraints 2** and **3** do not apply since no temporal expression and signal is present. The only available source of information is the commonsense knowledge on the basis of which can infer that $_{e2}$ stands in a precedence relation with $_{e1}$. In case we had more specific information (about the context of occurrence), the temporal relation could be overriden.

The analysis of the correlation between tense patterns and temporal relations has suggested that it is possible to associate a preference order for temporal relations according to the combination of the tense forms. In particular, it appears from the data that certain tense forms when appearing in particular tense patterns, like the *trapassato I* [past perfect], tend to grammaticalize particular temporal relations, while others are more prone to code a larger set of relations, like the *passato composto* [present perfect or simple past]. The preferences have been introduced to reduce the possible temporal relations which may be computed. To clarify how preference rules work consider the following example which is an adapatation from example 1. In this reduced and simplified formalization, t represent the beginning or ending point of the eventutalities, S the moment of utterance and R, as already stated a possible secondary deictic moment/point necessary to describe the semantics of some tense forms, like the *trapassato I.*

(2) Marco *è caduto*$_{e1}$. Giovanni l'*aveva spinto*$_{e2}$.
Marco fell$_{e1}$. *Giovanni had pushed him*$_{e2}$.
- discourse sequence: *passato composto*$_{e1}$ - *trapassato I*$_{e2}$
- $e_1 = ((E_1 \prec S) \wedge (t_1 \leq E_1 \leq t_2))$ [tense analysis for $_{e1}$]
- $e_2 = ((E_2 \prec R_2) \wedge (R_2 \prec S) \wedge (t_3 \leq R_2 \leq t_4))$ [tense analysis for $_{e2}$]
- $(t_1 \prec t_3) \wedge (t_2 \prec t_4) \wedge (t_1 \prec t_4) \wedge (t_2 ?? t_3))$
- possible temporal relations: $((E_2 \prec E_1) \vee (E_2\ m\ E_1) \vee (E_2\ o\ E_1))$

The final output does not provide a unique temporal relations due to the missing information between the ending point of e_1 and the beginning point of e_2, i.e. $(t_2??t_3)$. The application of the preference rule for sequences of *passato composto - trapassato I* states that the reliable temporal relation is that of precedence:

- possible temporal relations: $((E_2 \prec E_1) \vee (E_2\ m\ E_1) \vee (E_2\ o\ E_1))$
- Preference Rule: `if the sequence is` *passato composto - trapassato I*
  `then reduce the output to` $E_2 \prec E_1$
- final output: $(E_2 \prec E_1)$

It is important to point out that the preference rules do not apply for all tense patterns. For instance, with the *futuro composto* [future pefect] and the *futuro nel passato* [future-in-the-past], where the relationship between E and S cannot be reliably stated, no preference rules apply and the output of the component is obtained by disjunctive finely grained relations which can be rearranged in terms of coarse grained temporal relations.

Finally, it has been possible to formulate a saliency-based hierarchical order of application of the sources of information as reported in **Formula 1**. The symbol $\lessapprox$ stands for "in absence of more specific linguistic information, X is the most salient source of information" and $\lesssim$ for "in absence of more specific information, X is the most salient source of information". Notice that when stating "in absence of more specific (linguistic) information", we are referring to the constraints we have identified for the saliency of the sources of information:

Formula 1 (Hierarchical order of information) : COMMONSENSE KNOWLEDGE $\lessapprox$ (IMPLICIT SIGNALS $\lesssim$ TENSE $\lesssim$ VIEWPOINT ASPECT $\lesssim$ LEXICAL ASPECT $\lesssim$ TEMPORAL EXPRESSIONS $\lesssim$ EXPLICIT SIGNALS)

The saliency based hierarchy is an abstraction. A human interpreter always has at disposal all the sources of information. On the basis of the experimental data, we have deducted that the most probable order for processing the information is the one illustrated in the hierarchy since as soon as the subjects have found a reliable solution

they should have blocked their inferencing processes. The behavior of the pragmatic, i.e. commonsense, knowledge seems to offer further support to this observation. In fact, this source was selected as the most salient only when all the others were "absent", i.e. when the constraints we have illustrated were not respected.

3 The Model Architecture

This section is devoted to the illustration of the general architecture and mechanisms of a computational model for automatically resolve temporal relations in a text/discourse. The model is based on the empirical data and results illustrated above. Its modular organization is proposed as a strategy to improve the reliability of the output and avoid failure. Each module has some specialized functions and components which are conceived to deal with a source of information at time. The modules are organized on a pipeline according to which the output of one module represents the input for the following. On the basis of the hierarchy we have illustrated by means of the **Formula 1** and as a general strategy, the specialized components of each module should be activated only when necessary. Figure 1 illustrates the overall workflow of the model, from raw input text to the final output.

The first module is responsible for two primary tasks: the identification, normalization and assignment of temporal relations between the temporal expressions by means of a Timex Grammar and Normalizer and the identification of the eventualities and the assignment of their default lexical aspect through an Event Detector component. The two components take in input shallow parsed text since the chunks' extent approximates the extent of temporal expressions and eventualities. Moreover, chunks can be easily combined together for items whose extent corresponds to more than one token.

The second module has three main components which compute three different types of information strictly connected with each other, namely tense, viewpoint aspect and lexical aspect. The main result of the analysis of this internal submodule is the formalization of each eventuality in its corresponding interval representation. The output of these three components is necessary for two elements:

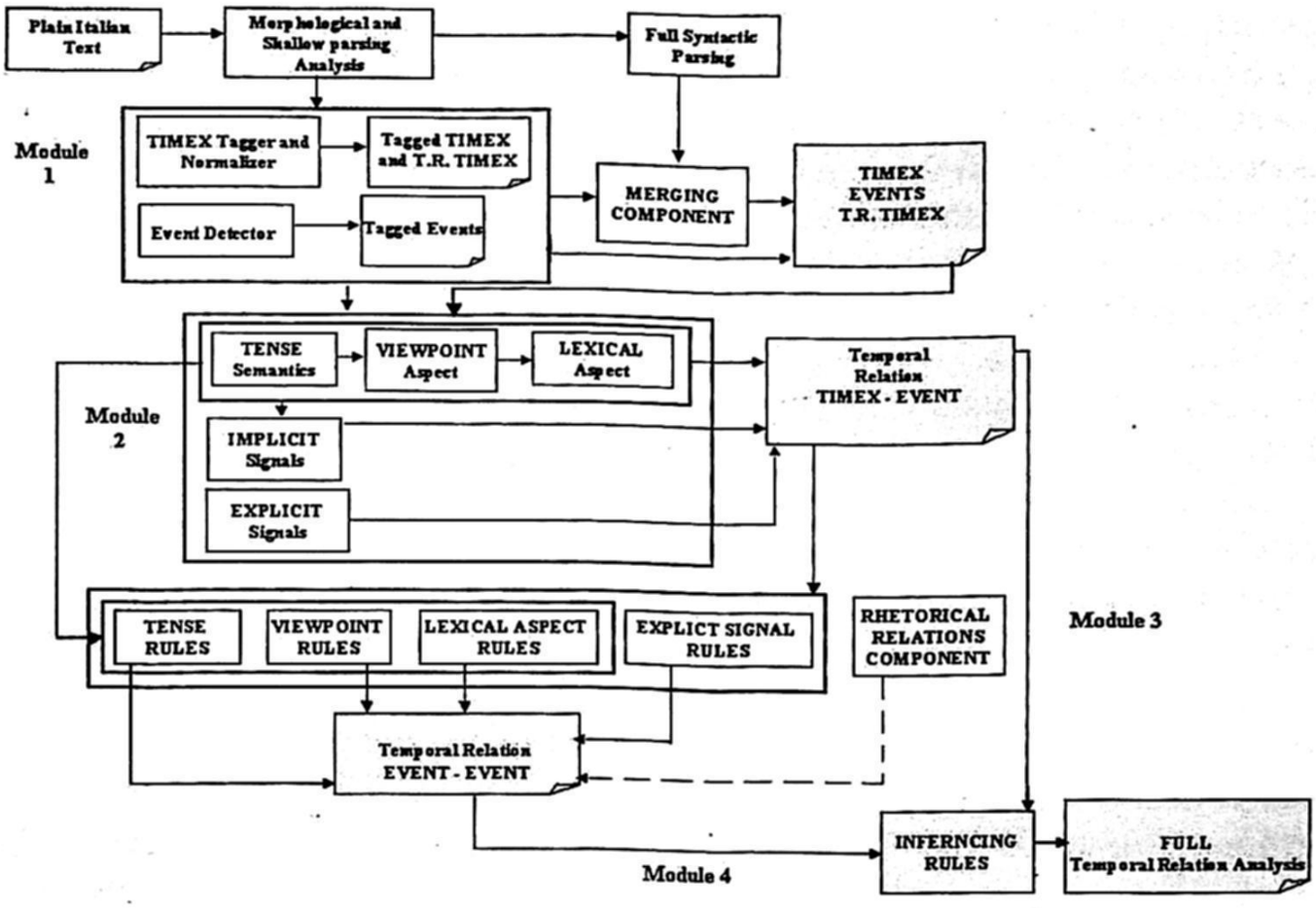

Fig. 1. Workflow of the Model.

firstly, it is used to to determine the temporal relations between temporal expressions and eventualities when associated to the output of the temporal expression component of module 1, and secondly, it is used to activate the components of module 3 only when they can provide a reliable output. The temporal relations which are assumed to be valid are all [5]'s 13 interval relations and [6]'s 8 instant-interval relations. The two signal components are activated when the connection between the eventuality and the temporal expression is "mediated" by a signal.

The third module is responsible for the identification of the temporal relations between eventualities. Each component has a set of heuristics based on the empirical data and provides as output the temporal relation value(s). The heuristics are divided into two main groups: one for complex sentence contexts and the others for adjacent eventualities in discourse segments. The four internal components are mutually exclusive one with respect to the other. A general

principle, which results from the constraints illustrated in sect. 2, guides their functioning. Each component is activated if and only if a set of preconditions is respected, otherwise the temporal ordering is completely inferred by means of the external component (module 4). The module assumes as basic temporal relations [5]'s 13 interval relations and [6]'s 8 instant-interval relations. A special predicate, *hold*, is postulated to account for the measure of the duration of the eventualities. As a general principle, the temporal relations between two adjacent eventualities are computed by considering the relations between the beginning and ending points of their interval representations. However, much of this information is missing or only vaguely present in the text/discourse. In order to deal with this issue, the final outputs of the internal components can differ in terms of the preciseness of the temporal knowledge expressed so that we can have (i.) precise temporal knowledge, when a single temporal relation can be stated or (ii.) coarse grained temporal knowledge, when more than one temporal relation can be inferred. In presence of coarse grained knowledge, the multiple temporal relations do not represent contradicting temporal representations, but related or conceptually adjacent temporal relations. Instead of expressing these types of temporal relations by means of disjunctive finely grained relations, we have decided to use of coarse grained knowledge based on [7]'s notion of conceptual neighbors. The main advantage of such a representation is two-folded: on the one hand, the model is somehow cognitively similar to the temporal representations that humans may have, and, on the other hand, it avoids that the inferencing module could fail to complete the whole set of temporal relations. According to our analysis, we have instances of precise temporal relations only when the output is obtained by three components, namely (i.) tense, (ii.) explicit signals and (iii.) discourse relations. The viewpoint and lexical aspect components will produce coarse grained temporal knowledge.

Module 4 is responsible for the inferencing process of temporal relations. This module takes in input both the output of module 2 and that from module 3 and it activates two different types of inferencing mechanisms according to which the module provides its output. When Module 2 provides the input, the eventualities are already connected by means of a temporal relation to a temporal expression. In this case, module 4 activates a set of inferencing rules according

to which the relations between the temporal expressions are transferred to their connected eventualities. Things are more complicate when the input comes from module 3. In this case, module 4 looks for couples of adjacent eventualities with one of them in common. Once identified, it will activate inferencing rules based on a transitivity table which preserves the insights of Allen's table and the coarse grained knowledge, as proposed by [7]. As for eventualities realized by nouns or other parts of speech different than verbs, our model implements a strategy based on [3]. Event nouns do not present information on their temporal location, thus the identification of the temporal relations requires an extended use of commonsense knowledge. Following [3]'s proposal, an abstract device, a Chronoscope, apply. This device allows temporal representation abstraction. The Chronoscope requires that we index temporal relations to a certain level of time granularity g. In our account, event nouns (and all parts-of-speech other than verbs) will be considered as simultaneous with the verbal eventualities with which they co-occur. This means that, the finely grained distinction between tensed eventualities will be maintained and preserved, and, at the same time, there is no need to make reference to commonsense knowledge in order to extend the model to event nouns. The only operation is to abstract their temporal representation on a level of same temporal granularity as that of the verbal eventualities.

4 Conclusion and Future Work

This paper illustrates a general architecture for robust temporal processing of text/discourse. The workflow of the model has been elaborated on the basis of experimetal data which have suggested a saliency-based order of application of the linguistic and non-linguistic sources of information involved in this task. With respect to previous research, this work has presented a first systematization of the linguistic devices involved in temporal processing and how they interact with each other. The development of a system compliant to the model has also advantages for the interpretation of errors and may facilitate their solution.

The actual relization of the model is ongoing. Some components (temporal expression tagger [8], an intrasentential tagger for tem-

poral relations between eventualities and temporal expressions in presence of signals [9]) and procedures (use of lexical resources for the identification of eventualities) have been realized and evaluated on specifically annotated documents by using different techiniques (rule-based systems *vs.* machine learning techniques). As a side effect of this work is the realization of an annotated corpus for Italian for events, temporal expressions and temporal relations, comparable to the English TimeBank[10].

As future work, we plan to experiment the validity of the experimental results by developing classifiers which apply different orders of application of the linguistic information with respect to those illustrated in sect. 2. A further element of analysis will be the evaluation of the flow of information from one module to the other and how errors can influence their functioning.

References

1. van der Meer, E., Beyer, R., Heinze, B., Badel, I.: Temporal order relations in language comprehension. Journal of Experimental Psychology: Learning, Memory and Cognition **28** (2002) 770–79
2. Kelter, S., Kaup, B., Claus, B.: Representing a described sequence of events: A dynamic view of narrative comprehension. Journal of Experimental Psychology: Learning, Memory and Cognition **30** (2004) 451–64
3. Mani, I.: *Chronoscopes*: A theory of underspecified temporal representation. In Schilder, F., Katz, G., Pustejovsky, J., eds.: Annotating, Extracting and Reasoning about Time and Events. LNAI. Springer-Verlag, Berlin Heidelberg (2007) 127–39
4. Forascu, C., Pistol, I., Cristea, D.: Temporality in relation with discourse structure. In: Proceedings of the Fifth International conference on Language Resources and Evaluation (LREC-06). (2006) 65–70
5. Allen, J.: Maintainig knowledge about temporal intervals. Communications of ACM **26**(11) (1983) 832–43
6. Allen, J., Hayes, P.: Moments and points in an interval-based temporal logic. Computational Intelligence **5**(3) (1989) 225–38
7. Freska, C.: Temporal reasoning based on semi-intervals. Artificial Intelligence **54** (1992) 199–227
8. Caselli, T., dell'Orletta, F., Prodanof, I.: A timeML compliant timex tagger for italian. In: Proceedings of the International Multiconference on Computer Science and Information Technology – IMCSIT 2009. Volume 4. (2009) 185 – 192
9. Caselli, T., dell'Orletta, F., Prodanof, I.: Temporal relations with signals: the case of italian temporal prepositions. In Lutz, C., Raskin, J.F., eds.: 16th International Symposium on Temporal Representation and Reasoning, 2009. TIME 2009. (2009) 125 – 132
10. Pustejovsky, J., Hanks, P., Saurí, R., See, A., Gaizauskas, R., Setzer, A., Radev, D., Sundheim, B., Day, D., Ferro, L., Lazo, M.: The TIMEBANK corpus. In: Corpus Linguistics 2003. (2003)

Near-Synonym Choice using a 5-gram Language Model

Aminul Islam and Diana Inkpen

University of Ottawa
School of Information Technology and Engineering
Ottawa, ON, Canada, K1N 6N5
{mdislam, diana}@site.uottawa.ca

Abstract. In this work, an unsupervised statistical method for automatic choice of near-synonyms is presented and compared to the state-of-the-art. We use a 5-gram language model built from the Google Web 1T data set. The proposed method works automatically, does not require any human-annotated knowledge resources (e.g., ontologies) and can be applied to different languages. Our evaluation experiments show that this method outperforms two previous methods on the same task. We also show that our proposed unsupervised method is comparable to a supervised method on the same task. This work is applicable to an intelligent thesaurus, machine translation, and natural language generation.

1 Introduction

Choosing the wrong near-synonym can convey unwanted connotations, implications, or attitudes. In machine translation and natural language generation systems, the choice among near-synonyms needs to be made carefully. By *near-synonyms* we mean words that have the same meaning, but differ in lexical nuances. For example, *error*, *mistake*, and *blunder* all mean a generic type of error, but *blunder* carries an implication of *accident* or *ignorance*. In addition to paying attention to lexical nuances, when choosing a word we need to make sure it fits well with the other words in a sentence. In this paper we investigate how the collocational properties of near-synonyms can help with choosing the best words. This problem is difficult because the near-synonyms have senses that are very close to each other, and therefore they occur in similar contexts. We build a strong representation of the context in order to capture the more subtle differences specific to each near-synonym.

The work we present here can be used in an intelligent thesaurus. A writer can access a thesaurus to retrieve words that are similar to a given word, when there is a need to avoid repeating the same word, or when the word does not seem to be the best choice in the context. A standard thesaurus does not offer any explanation about the differences in nuances of meaning between the possible word choices.

This work can also be applied to a natural language generation system [1] that needs to choose among near-synonyms. Inkpen and Hirst [1] included a

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 41-52

Received 23/11/09
Accepted 16/01/10
Final version 10/03/10

preliminary collocation module that reduces the risk of choosing a near-synonym that does not fit with the other words in a generated sentence (i.e., violates collocational constraints). The work presented in this paper allows for a more comprehensive near-synonym collocation module.

The task we address in this paper is the selection of the best near-synonym that should be used in a particular context. Inkpen [2] argues that the natural way to validate an algorithm for this task would be to ask human readers to evaluate the quality of the algorithm's output, but this kind of evaluation would be very laborious. Instead, Inkpen [2] validates her algorithms by deleting selected words from sample sentences, to see whether the algorithms can restore the missing words. That is, she creates a *lexical gap* and evaluates the ability of the algorithms to fill the lexical gap. Two examples from [2] are presented in Figure 1. All the near-synonyms of the original word, including the word itself, become the choices in the solution set (see the figure for two examples of solution sets). The task is to automatically fill the gap with the best choice in the particular context. We present a method that can be used to scoring the choices. For our particular task, we choose only the highest scoring near-synonym. In order to evaluate how well our method works we consider that the only correct solution is the original word. This will cause our evaluation scores to underestimate the performance of our method, as more than one choice will sometimes be a perfect solution. Moreover, what we consider to be the best choice is the typical usage in the corpus, but it may vary from writer to writer. Nonetheless, it is a convenient way of producing test data in an automatic way. To verify how difficult the task is for humans, Inkpen [2] performed experiments with human judges on a sample of the test data.

The near-synonym choice method that we propose here uses the Google Web 1T n-gram data set [3], contributed by Google Inc., that contains English word n-grams (from unigrams to 5-grams) and their observed frequency counts calculated over 1 trillion words from web page text collected by Google in January 2006. The text was tokenized following the Penn Treebank tokenization, except that hyphenated words, dates, email addresses and URLs are kept as single tokens. The sentence boundaries are marked with two special tokens <S> and </S>. Words that occurred fewer than 200 times were replaced with the special token <UNK>. Table 1 shows the data sizes of the Web 1T corpus. The n-grams

Table 1. Google Web 1T Data Sizes

Number of	Number	Size on disk (in KB)
Tokens	1,024,908,267,229	N/A
Sentences	95,119,665,584	N/A
Unigrams	13,588,391	185,569
Bigrams	314,843,401	5,213,440
Trigrams	977,069,902	19,978,540
4-grams	1,313,818,354	32,040,884
5-grams	1,176,470,663	33,678,504

themselves must appear at least 40 times to be included in the Web 1T corpus[1]. It is expected that this data will be useful for statistical language modeling, e.g., for machine translation or speech recognition, as well as for other uses.

Sentence: This could be improved by more detailed consideration of the processes of ..... propagation inherent in digitizing procedures.
Original near-synonym: error
Solution set: mistake, blooper, blunder, boner, contretemps, error, faux pas, goof, slip, solecism
Sentence: The day after this raid was the official start of operation strangle, an attempt to completely destroy the ..... lines of communication.
Original near-synonym: enemy
Solution set: opponent, adversary, antagonist, competitor, enemy, foe, rival

---

Fig. 1. Examples of sentences with a lexical gap, and candidate near-synonyms to fill the gap.

This paper is organized as follow: Section 2 presents a brief overview of the related work. Our proposed method is described in Section 3. Evaluation and experimental results are discussed in Section 4. We conclude in Section 5.

2 Related Work

The idea of using the Google Web 1T n-gram data set as a resource in different natural language processing applications has been exploited by many researchers. Islam and Inkpen [4] use 3-grams of this data set to detect and correct real-word spelling errors and also use n-grams to only correct real-word spelling errors [5]. Nulty and Costello [6] deduce the semantic relation that holds between two nouns in a noun-noun compound phrase such as "flu virus" or "morning exercise" using lexical patterns in the Google Web 1T corpus. Klein and Nelson [7] investigate the relationship between *term count* (*TC*) and *document frequency* (*DF*) values of terms occurring in the Web as Corpus (WaC) and also the similarity between *TC* values obtained from the WaC and the Google n-gram dataset and they mention that a strong correlation between the two would give them confidence in using the Google n-grams to estimate accurate inverse document frequency (IDF) values in order to generate well-performing lexical signatures based on the TF-IDF scheme. Murphy and Curran [8] explore the strengths and limitations of Mutual Exclusion Bootstrapping (MEB) by applying it to two novel lexical-semantic extraction tasks: extracting bigram named entities and WordNet lexical file classes [9] from the Google Web 1T 5-grams.

Turney *et al.* [10] addressed the multiple-choice synonym problem: given a word, choose a synonym for that word, among a set of possible solutions. In

[1] Details of the Google Web 1T data set can be found at www.ldc.upenn.edu/Catalog/docs/LDC2006T13/readme.txt

this case the solutions contain one synonym and some other (unrelated) words. They achieves high performance by combining classifiers. Clarke and Terra [11] addressed the same problem as Turney *et al.*, using statistical associations measures computed with counts from the Waterloo terabyte corpus. In our case, all the possible solutions are synonyms of each other, and the task is to choose one that best matches the context: the sentence in which the original synonym is replaced with a gap. It is much harder to choose between words that are near-synonyms because the context features that differentiate a word from other words might be shared among the near-synonyms.

In fact, the works that address exactly the same task are that of Edmonds [12] and Inkpen [2], as far as we are aware. Edmonds [12] gives a solution based on a lexical co-occurrence network that included second-order co-occurrences whereas Inkpen [2] uses a much larger corpus and a simpler method, and obtains better results than that of [12].

Inkpen's [2] unsupervised method is based on the mutual information scores between a near-synonym and the content words in the context filtering out the stopwords[2]. The *pointwise mutual information* (*PMI*) between two words x and y compares the probability of observing the two words together (their joint probability) to the probabilities of observing x and y independently (the probability of occurring together by chance) [13].

$$PMI(x,y) = \log_2 \frac{P(x,y)}{P(x)P(y)}$$

The probabilities can be approximated by: $P(x) = C(x)/N$, $P(y) = C(y)/N$, $P(x,y) = C(x,y)/N$, where C denote frequency counts and N is the total number of words in the corpus. Therefore,

$$PMI(x,y) = \log_2 \frac{C(x,y) \cdot N}{C(x) \cdot C(y)}$$

where N can be ignored in comparisons, since is it the same in all the cases. Inkpen [2] models the context as a window of size $2k$ around the gap (the missing word): k words to the left and k words to the right of the gap. If the sentence is $s = \cdots w_1 \cdots w_k \;\; Gap \;\; w_{k+1} \cdots w_{2k} \cdots$, for each near-synonym NS_i from the group of candidates, the score is computed by the following formula:

$$Score(NS_i, s) = \Sigma_{j=1}^{k-1} PMI(NS_i, w_j) \; + \Sigma_{j=k+1}^{2k} PMI(NS_i, w_j).$$

In a supervised learning method, Inkpen [2] trains classifiers for each group of near-synonyms. The classes are the near-synonyms in the solution set. Each sentence is converted into a vector of features to be used for training the supervised classifiers. Inkpen used two types of features. One type of features consists in the scores of the left and right context with each class (i.e., with each near-synonym from the group). The number of features of this type is equal to twice the number of classes: one feature for the score between the near-synonym and

[2] We do not filter out stopwords or punctuation in our method.

the part of the sentence at the left of the gap, and one feature for the score between the near-synonym and the part of the sentence at the right of the gap. The second type of features is formed by the words in the context windows. For each group of near-synonyms, Inkpen used as features the 500 most frequent words situated close to the gaps in a development set. The value of a word feature for each training example is 1 if the word is present in the sentence (at the left or at the right of the gap), and 0 otherwise. Inkpen trained classifiers using several machine learning algorithms to see which one is best at discriminating among the near-synonyms.

There has been quite a lot of work in unsupervised learning of word clusters based on n-grams [14–16]. Our task has similarities to the word sense disambiguation task. Our near-synonyms have senses that are very close to each other. In Senseval, some of the fine-grained senses are also close to each other, so they might occur in similar contexts, while the coarse-grained senses are expected to occur in distinct contexts. In our case, the near-synonyms are different words to choose from, not the same word with different senses.

3 Proposed Method

Our task is to find the best near-synonym from a set of candidates that could fill in the gap in an input text, using the Google Web 1T data set. Let us consider an input text W which after tokenization[3] has p ($2 \leq p \leq 9$) words[4], i.e.,

$$W = \{\ldots\ w_{i-4}\ w_{i-3}\ w_{i-2}\ w_{i-1}\ \boxed{w_i}\ w_{i+1}\ w_{i+2}\ w_{i+3}\ w_{i+4}\ \ldots\}$$

where $\boxed{w_i}$ (in position i) indicates the gap and w_i in $\boxed{w_i}$ denotes a set of m near-synonyms (i.e., $w_i = \{s_1, s_2, \cdots, s_j, \cdots, s_m\}$). We take into account at most four words before the gap and at most four words after the gap. Our task is to choose the $s_j \in w_i$ that best matches with the context. In other words, the position i is the gap that needs to be filled with the best suited member from the set, w_i.

[3] We need to tokenize the input sentence to make the n-grams formed using the tokens returned after the tokenization consistent with the Google n-grams. The input sentence is tokenized in a manner similar to the tokenization of the Wall Street Journal portion of the Penn Treebank. Notable exceptions include the following:

- Hyphenated word are usually separated, and hyphenated numbers usually form one token.
- Sequences of numbers separated by slashes (e.g., in dates) form one token.
- Sequences that look like urls or email addresses form one token.

[4] If the input text has more than 9 words then we keep at most four words before the gap and at most four words after the gap to make the length of the text 9. We choose these numbers so that we could maximize the number of n-grams to use, given that we have up to 5-grams in the Google Web 1T data set.

We construct m strings $(S_1 \cdots S_m)$ replacing the gaps in position i with $s_j \in w_i$ as follows:

$$S_1 = \cdots w_{i-4} w_{i-3} w_{i-2} w_{i-1} s_1 w_{i+1} w_{i+2} w_{i+3} w_{i+4} \cdots$$

$$S_2 = \cdots w_{i-4} w_{i-3} w_{i-2} w_{i-1} s_2 w_{i+1} w_{i+2} w_{i+3} w_{i+4} \cdots$$

$$\vdots$$

$$S_m = \cdots w_{i-4} w_{i-3} w_{i-2} w_{i-1} s_m w_{i+1} w_{i+2} w_{i+3} w_{i+4} \cdots$$

Using equation 7 in Section 3.2, we calculate $P(S_1), \cdots, P(S_m)$. The index of the target synonym, j, will be $\underset{j \in 1 \cdots m}{\operatorname{argmax}} P(S_j)$.

3.1 n-gram Language Model

A *language model* is usually formulated as a probability distribution $P(S)$ over strings S, and attempts to reflect how frequently a string S occurs as a sentence. The most widely-used language models, by far, are n-gram language models [17]. We introduce these models by considering the case $n = 5$; these models are called 5-gram models. For a sentence S composed of the words $w_1 \cdots w_p$, without loss of generality we can express $P(S)$ as

$$\begin{aligned} P(S) &= P(w_1)P(w_2|w_1)P(w_3|w_1 w_2) \cdots P(w_p|w_1 \cdots w_{p-1}) \\ &= \prod_{i=1}^{p} P(w_i|w_1 \cdots w_{i-1}) \end{aligned} \tag{1}$$

For n-gram models where $n > 2$, we condition the probability of a word on the identity of the last n-1 words. Generalizing equation 1 to $n > 2$, we get

$$P(S) = \prod_{i=1}^{p+1} P(w_i|w_{i-n+1}^{i-1}) \tag{2}$$

where w_i^j denotes the words $w_i \cdots w_j$. To estimate $P(w_i|w_{i-n+1}^{i-1})$, the frequency with which the word w_i occurs given that the words w_{i-n+1}^{i-1} precede the current word w_i, we simply count how often the n-gram w_{i-n+1}^{i} occurs in some text and normalize by the total number of occurrences of any word in the same context. Let $C(w_{i-n+1}^{i})$ denote the number of times the n-gram w_{i-n+1}^{i} occurs in the given text. Then

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i})}{\sum_{w_i} C(w_{i-n+1}^{i})} \tag{3}$$

Notice that $C(w_{i-n+1}^{i-1}) \geq \sum_{w_i} C(w_{i-n+1}^{i})$. To understand the inequality of $C(w_{i-n+1}^{i-1})$ and $\sum_{w_i} C(w_{i-n+1}^{i})$, assume that both i and n are 5. Then, $C(w_{i-n+1}^{i-1})$ becomes $C(w_1 w_2 w_3 w_4)$, which is actually the frequency of a specific 4-gram, $w_1 w_2 w_3 w_4$, and $\sum_{w_i} C(w_{i-n+1}^{i})$ becomes $\sum_{w_5} C(w_1 w_2 w_3 w_4 w_5)$, which is the

sum of the frequencies of all the 5-grams that start with the 4-gram, $w_1w_2w_3w_4$. Thus, in general, $C(w_1w_2w_3w_4)$ is equal to $\sum_{w_5} C(w_1w_2w_3w_4w_5)$. But, for some specific cases, it is possible that $C(w_1w_2w_3w_4)$ is greater than $\sum_{w_5} C(w_1w_2w_3w_4w_5)$. For example, all the n-grams ($2 \leq n \leq 5$) that appear less than 40 times have been filtered out from the Google Web 1T n-grams. This means all the 5-grams (starting with the 4-gram $w_1w_2w_3w_4$) that appear less than 40 times have not been included in $\sum_{w_5} C(w_1w_2w_3w_4w_5)$. Thus, when we deal with the Web 1T 5-grams and 4-grams, it is obvious that $C(w_1w_2w_3w_4)$ is greater than or equal to $\sum_{w_5} C(w_1w_2w_3w_4w_5)$. Thus, in general, we can say that $C(w_{i-n+1}^{i-1}) \geq \sum_{w_i} C(w_{i-n+1}^{i})$. This also supports the idea of using the missing count (equation 4) in the smoothing formula (equation 5).

We use a smoothing method loosely based on the *one-count* method described in [18]. Because tokens that appears less than 200 times and n-grams that appear less than 40 times have been filtered out from the Web 1T, we use n-grams with missing counts instead of n-grams with one counts [19]. The missing count is defined as:

$$M(w_{i-n+1}^{i-1}) = C(w_{i-n+1}^{i-1}) - \sum_{w_i} C(w_{i-n+1}^{i}) \tag{4}$$

The corresponding smoothing formula is:

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i}) + (1+\alpha_n)M(w_{i-n+1}^{i-1})P(w_i|w_{i-n+2}^{i-1})}{C(w_{i-n+1}^{i-1}) + \alpha_n M(w_{i-n+1}^{i-1})} \tag{5}$$

Yuret [19] optimized the parameters $\alpha_n > 0$ for $n = 2 \cdots 5$ on the Brown corpus to yield a cross entropy of 8.06 bits per token. The optimized parameters are: $\alpha_2 = 6.71$, $\alpha_3 = 5.94$, $\alpha_4 = 6.55$, $\alpha_5 = 5.71$

Thus, incorporating the smoothing formula in equation 2, we get

$$P(S) = \prod_{i=1}^{p+1} \frac{C(w_{i-n+1}^{i}) + (1+\alpha_n)M(w_{i-n+1}^{i-1})P(w_i|w_{i-n+2}^{i-1})}{C(w_{i-n+1}^{i-1}) + \alpha_n M(w_{i-n+1}^{i-1})} \tag{6}$$

3.2 The Language Model Used for Our Task

For the specified task, we simplify the 5-gram model described in Section 3.1. From equation 2, it is clear that the maximum number of products possible is 10 as $2 \leq p \leq 9$. Among these products, we can omit the products $P(w_{i-1}|w_{i-5}^{i-2})$, $P(w_{i-2}|w_{i-6}^{i-3})$, $P(w_{i-3}|w_{i-7}^{i-4})$, $P(w_{i-4}|w_{i-8}^{i-5})$, and $P(w_{i+5}|w_{i+1}^{i+4})$ because these product items have the same values for all $j \in 1 \cdots m$. Thus, the five product items that we consider are: $P(w_i|w_{i-4}^{i-1})$, $P(w_{i+1}|w_{i-3}^{i})$, $P(w_{i+2}|w_{i-2}^{i+1})$, $P(w_{i+3}|w_{i-1}^{i+2})$, and $P(w_{i+4}|w_{i}^{i+3})$. Applying this simplification in equation 2 and equation 6, we get

$$\begin{aligned} P(S) &= \prod_{i=5}^{p} P(w_i|w_{i-n+1}^{i-1}) \\ &= \prod_{i=5}^{p} \frac{C(w_{i-n+1}^{i}) + (1+\alpha_n)M(w_{i-n+1}^{i-1})P(w_i|w_{i-n+2}^{i-1})}{C(w_{i-n+1}^{i-1}) + \alpha_n M(w_{i-n+1}^{i-1})} \end{aligned} \tag{7}$$

Equation 7 is actually used in a recursive way for n=5,4,3,2,1 (i.e., if the current n-gram count is zero, then it is used with a lower n-gram, and so on). For example, $P(w_5|w_1w_2w_3w_4)$ is a function of $C(w_1w_2w_3w_4w_5)$ and $P(w_5|w_2w_3w_4)$; if $C(w_1w_2w_3w_4w_5) > 0$ then we do not consider $P(w_5|w_2w_3w_4)$. This is a backoff language model.

4 Evaluation and Experimental Results

4.1 Comparison to Edmonds's and Inkpen's methods

In this section we present results of the proposed method explained in Section 3. We compare our results with those of Inkpen [2] and Edmonds [12]. Edmonds [12] solution used the texts from the year 1989 of the Wall Street Journal (WSJ) to build a lexical co-occurrence network for each of the seven groups of near-synonyms from Table 2. The network included second-order co-occurrences. Edmonds used the WSJ 1987 texts for testing, and reported accuracies only a little higher than the baseline. Inkpen's [2] method is based on mutual information, not on co-occurrence counts. Inkpen's counts are collected from a much larger corpus.

Test Set	Number of Cases	Accuracy				
		Base Line	Edmonds's Method	Inkpen's Method (Supervised)	Inkpen's Method (Unsup.)	Proposed Method (Unsup.)
difficult, hard, tough	6,630	41.7%	47.9%	57.3%	59.1%	**63.2%**
error, mistake, oversight	1,052	30.9%	48.9%	70.8%	61.5%	**78.7%**
job, task, duty	5,506	70.2%	68.9%	**86.7%**	73.3%	78.2%
responsibility, burden, obligation, commitment	3,115	38.0%	45.3%	66.7%	66.0%	**72.2%**
material, stuff, substance	1,715	59.5%	64.6%	71.0%	**72.2%**	70.4%
give, provide, offer	11,504	36.7%	48.6%	**56.1%**	52.7%	55.8%
settle, resolve	1,594	37.0%	65.9%	75.8%	**76.9%**	70.8%
ALL (average over all sentences)	31,116	44.9%	53.5%	65.2%	61.7%	**65.3%**
ALL (average from group averages)	31,116	44.8%	55.7%	69.2%	66.0%	**69.9%**

Table 2. Comparison among the new proposed method, a baseline algorithm, Edmonds's method, and Inkpen's unsupervised and supervised method

For comparison purposes, in this section we use the same test data (WSJ 1987) and the same groups of near-synonyms. The seven groups of near-synonyms used by Edmonds are listed in the first column of Table 2. The near-synonyms in the seven groups were chosen to have low polysemy. This means that some sentences with wrong senses of near-synonyms might be in the automatically produced test set, but hopefully not many.

Before we look at the results, we mention that the accuracy values we compute are the percentage of correct choices when filling in the gap with the winning near-synonym. The expected solution is the near-synonym that was originally in the sentence, and it was taken out to create the gap. This measure is conservative; it does not consider cases when more than one solution is correct.

Table 2 presents the comparative results for the seven groups of near-synonyms. The second last row averages the accuracies for all the test sentences, i.e., these are calculated as the number of correct choices returned over total number of sentences (i.e., 31116). The last row averages the accuracies for all the groups averages, i.e., these are calculated as the sum of the accuracies (in percentage) of all the seven groups over the number of groups (i.e., 7). The second column shows how many test sentences we collected for each near-synonym group. The third column is for the baseline algorithm that always chooses the most frequent near-synonym. The fourth column presents the results reported in [12]. The fifth column presents the result of Inkpen's [2] supervised method when using boosting (decision stumps) as machine learning method and PMI+500 words as features. The sixth column presents the result of Inkpen's [2] unsupervised method when using word counts in PMI, and the last column is for our proposed unsupervised method using the Google n-grams. We show in bold the best accuracy figure for each data set. We notice that the automatic choice is more difficult for some near-synonym groups than for the others.

Our method performs significantly better than the baseline algorithm, Edmond's method, and Inkpen's unsupervised method and comparable to Inkpen's supervised method. For all the results presented in this paper, statistical significance tests were done using the paired t-test, as described in [20], page 209. Error analysis reveals that incorrect choices happen more often when the context is weak, that is, very short sentences or sentences with very few content words.

On average, our method performs 25 percentage points better than the baseline algorithm, 14 percentage points better than Edmonds's method, 4 percentage points better than Inkpen's unsupervised method, and comparable to Inkpen's supervised method. An important advantage of our method is that it works on any group of near-synonyms without training, whereas Edmonds's method requires a lexical co-occurrence network to be built in advance for each group of near-synonyms and Inkpen's supervised method requires training for each near-synonym group.

Some examples of correct and incorrect choices, using our proposed method are shown in Table 3. Table 4 shows some examples of cases where our proposed method fails to generate any suggestion. Cases where our method failed to provide any suggestion are due to the appearances of some very uncommon proper names or nouns, contractions (e.g., n't), hyphens between two words (e.g., teen-agers), single and double inverted commas in the n-grams, and so on. Preprocessing to tackle this issues would improve the results.

CORRECT CHOICE:
... viewed widely as a *mistake* → mistake [error, mistake, oversight] and a major ...
... analysts expect stocks to *settle* → settle [settle, resolve] into a steady ...
... Sometimes that *task* → task [job, task, duty] is very straightforward ...
... carry a heavier tax *burden* → burden [responsibility, burden, obligation, commitment] during 1987 because ...
INCORRECT CHOICE:
... would be a political *mistake* → error [error, mistake, oversight] to criticize the ...
... its energies on the *material* → substance [material, stuff, substance] as well as ...
... 23 , and must *provide* → give [give, provide, offer] Washington-based USAir at ...
... Phog Allen – to *resolve* → settle [settle, resolve] the burning question ...

Table 3. Examples of correct and incorrect choices using our proposed method. Italics indicate the proposed near-synonym choice returned by the method, arrow indicates the original near-synonym, square brackets indicate the solution set.

NO CHOICE:
... two exchanges ' " → commitment [responsibility, burden, obligation, commitment] to making serious ...
... He said ECD 's → material [material, stuff, substance] is a multicomponent ...
... Safe Rides , teen-agers → give [give, provide, offer] rides to other ...
... guilty plea does n't → resolve [settle, resolve] the issue for ...
... sees Mr. Haig 's → tough [difficult, hard, tough] line toward Cuba ...
... The 1980 Intelligence → Oversight [error, mistake, oversight] Act requires that ...
... thought Mr. Cassoni 's → job [job, task, duty] will be made ...

Table 4. Examples of sentences where our method fails to generate any suggestion.

4.2 Experiment with human judges

Inkpen [2] asked two human judges, native speakers of English, to guess the missing word in a random sample of the experimental data set (50 sentences for each of the 7 groups of near-synonyms, 350 sentences in total). The results in Table 5 show that the agreement between the two judges is high (78.5%), but not perfect. This means the task is difficult even if some wrong senses in the automatically-produced test data might have made the task easier in a few cases.

[5] Here, each of the seven groups has equal number of sentences, which is 50. Thus, the average from all 350 sentences and the average from group averages are the same.

Test set	J1-J2 Agreement	J1 Accuracy	J2 Accuracy	Inkpen's Accuracy	Our Accuracy
difficult, hard, tough	72%	70%	76%	53%	62%
error, mistake, oversight	82%	84%	84%	68%	70%
job, task, duty	86%	92%	92%	78%	80%
responsibility, burden, obligation, commitment	76%	82%	76%	66%	76%
material, stuff, substance	76%	82%	74%	64%	56%
give, provide, offer	78%	68%	70%	52%	52%
settle, resolve	80%	80%	90%	77%	66%
All (average)[5]	78.5%	79.7%	80.2%	65.4%	66%

Table 5. Experiments with two human judges on a random subset of the experimental data set

The human judges were allowed to choose more than one correct answer when they were convinced that more than one near-synonym fits well in the context. They used this option sparingly, only in 5% of the 350 sentences. In future work, we plan to allow the system to make more than one choice when appropriate (e.g., when the second choice has a very close score to the first choice).

5 Conclusions

We presented an unsupervised statistical method of choosing the best near-synonym in a context. We compared this method with three previous methods (Edmonds's method and two of Inkpen's method) and show that the performance improved considerably. It is interesting that our unsupervised statistical method is comparable to a supervised learning method.

Future work includes an intelligent thesaurus and a natural language generation system that has knowledge of nuances of meaning of near-synonyms. We plan to include a near-synonym sense disambiguation module to ensure that the thesaurus does not offer alternatives for wrong senses of words.

References

1. Inkpen, D.Z., Hirst, G.: Near-synonym choice in natural language generation. In: Proceedings of the International Conference RANLP-2003 (Recent Advances in Natural Language Processing), Borovets, Bulgaria (2003) 204–211
2. Inkpen, D.: A statistical model for near-synonym choice. ACM Transactions on Speech and Language Processing 4 (2007) 1–17
3. Brants, T., Franz, A.: Web 1T 5-gram corpus version 1.1. Technical report, Google Research (2006)

4. Islam, A., Inkpen, D.: Real-word spelling correction using Google Web 1T 3-grams. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, Singapore, Association for Computational Linguistics (2009) 1241–1249
5. Islam, A., Inkpen, D.: Real-word spelling correction using Google Web 1T n-gram data set. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, ACM (2009) 1689–1692
6. Nulty, P., Costello, F.: Using lexical patterns in the Google Web 1T corpus to deduce semantic relations between nouns. In: Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions (SEW-2009), Boulder, Colorado, Association for Computational Linguistics (2009) 58–63
7. Klein, M., Nelson, M.L.: Correlation of term count and document frequency for Google n-grams. In et al., B.M., ed.: Proceedings of the 31st European Conference on Information Retrieval. Volume 5478/2009 of Lecture Notes in Computer Science., Springer Berlin / Heidelberg (2009) 620–627
8. Murphy, T., Curran, J.: Experiments in mutual exclusion bootstrapping. In: Proceedings of the Australasian Language Technology Workshop 2007, Melbourne, Australia (2007) 66–74
9. Fellbaum, C., ed.: WordNet: An electronic lexical database. MIT Press (1998)
10. Turney, P., Littman, M., Bigham, J., Shnayder, V.: Combining independent modules to solve multiple-choice synonym and analogy problems. In: Proceedings of the International Conference RANLP-2003 (Recent Advances in Natural Language Processing), Borovets, Bulgaria (2003) 482–489
11. Clarke, C.L.A., Terra, E.: Frequency estimates for statistical word similarity measures. In: Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003), Edmonton, Canada (2003) 165–172
12. Edmonds, P.: Choosing the word most typical in context using a lexical co-occurrence network. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, Madrid, Spain (1997) 507–509
13. Church, K., Hanks, P.: Word association norms, mutual information and lexicography. Computational Linguistics **16 (1)** (1991) 22–29
14. Brown, P.F., deSouza, P.V., Mercer, R.L., Pietra, V.J.D., Lai, J.C.: Class-based *n*-gram models of natural language. Computational Linguistics **18** (1992) 467–479
15. Pereira, F., Tishby, N., Lee, L.: Distributional clustering of english words. In: Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics. (1993) 183–190
16. Lin, D.: Automatic retrieval and clustering of similar words. In: Proceedings of the 17th international conference on Computational linguistics, Morristown, NJ, USA, Association for Computational Linguistics (1998) 768–774
17. Chen, S.F., Goodman, J.T.: An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Computer Science Group, Harvard University (1998)
18. Chen, S.F., Goodman, J.: An empirical study of smoothing techniques for language modeling. In: 34th Annual Meeting of the Association for Computational Linguistics. (1996) 310–318
19. Yuret, D.: KU: Word sense disambiguation by substitution. In: Proceedings of the 4th workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic (2007) 207–214
20. Manning, C., Schütze, H.: Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge, Massachusetts (1999)

Morphology, Syntax, Named Entity Recognition

Exploring the *N*-th Dimension of Language

Prakash Mondal

Cognitive Science Lab, International Institute of Information Technology

Gachibowli, Hyderabad 500032, India

Abstract. This paper is aimed at exploring the hidden fundamental computational property of natural language that has been so elusive that it has made all attempts to characterize its real computational property ultimately fail. Earlier natural language was thought to be context-free. However, it was gradually realized that this does not hold much water given that a range of natural language phenomena have been found as being of non-context-free character that they have almost scuttled plans to brand natural language context-free. So it has been suggested that natural language is mildly context-sensitive and to some extent context-free. In all, it seems that the issue over the exact computational property has not yet been solved. Against this background it will be proposed that this exact computational property of natural language is perhaps the *N*-th dimension of language, if what we mean by dimension is nothing but universal (computational) property of natural language.

Keywords: Hidden fundamental variable; natural language; context-free; computational property; *N*-th dimension of language.

1 Introduction

Let's start with the question "What exactly is the universal computational property of natural language?" The simple and perhaps a little mysterious answer is that nobody knows what it is. Then we may ask the other natural and subsequent question "Why?". Even here we are nowhere nearer to having any clearer grasp of the reason why the exact natural language computational property is beyond our reach. Perhaps it is not knowable at all. Or perhaps it is knowable, but the question on this issue is not a relevant one. Or even perhaps there is no single answer to this question. This is the reason why we have till now a plethora of grammar formalisms or models of natural language defining and characterizing different classes of language; some defining context-free languages, some context-sensitive, and some are of mixed nature with each type having weak or strong generative capacity. Here in this paper it will be argued that this seemingly unknowable computational fundamental having universal applicability and realization constitutes the *N*-th dimension of natural language. Hence, we may know a little bit about *N-1,...,N-m* (when *m* is an arbitrary

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 55-66

Received 28/10/09
Accepted 16/01/10
Final version 08/03/10

number and $m < N$) computational properties of natural language, but we do not understand the N-th. This is the N-th dimension of natural language in that we do not yet know what N stands for. On the surface of it all, it appears that the exact computational property of natural language defies any attempt to uncover it.

However, it will also be argued that it may have, though not necessarily, something to do with the emergent nature of natural language since natural language is an emergent entity derived out of the interaction with several cognitive domains involving emotion, social cognition, vision, memory, attention, motor system, auditory system etc. And here at the linguistic level, language emerges through integrated and interwoven, but partially constrained, interactions between syntax, semantics, morphology, lexicon and phonology which form an overlapping network. This is a case of recursive emergence in that there are two layers of emergence: one at the level of language and another at the level of cognitive architecture as a whole.

Apart from all that, it will also be shown that as we do not know the N-th dimension of natural language, it hampers our progress in natural language processing in particular and artificial intelligence in general. That means we do not yet have a universally implementable grammar formalism or formal model of natural language that can characterize to the fullest possible extent *only and all* conceivable natural languages and all possible natural language expressions in some sense but metaphorically like Universal Turing machine. The lack of this hinders us in building a parser that will have in it such a grammar as being able to universally generate all possible natural languages and all possible natural language expressions. The same can be said of natural language generation system, natural language understanding system etc. However, it does not now exist, so it does not necessarily entail that we can never achieve this. Once achieved, it will perhaps serve as a key to towards solving most of our problems we are facing in building a universally implementable language model that may facilitate cost-effective solutions across languages. Let's now explore the possible nature, form and origin of the N-th dimension of language.

2 Laying the Exploration out against the Backdrop

From the earliest days of Chomskyan hierarchy, the computational nature of natural language was assumed to be context-free. In fact, Chomsky's generative grammar started out with context-free grammar [1]. Even before that, language was characterized as context-free in the structuralist tradition. But later Chomsky [2] himself argued against the validity of context-free grammar in characterizing natural language. And with this the debate started about whether natural language is context-free or not. With this a cascade of papers came off arguing that natural language cannot be context-free [3], [4], [5], [6], [7], [8]. But Pullum and Gazdar [9] and Pullum [10] on the other hand threw skepticism over such arguments. In the middle, Joshi [11], [12] tried to strike a middle ground between mild context-sensitivity and context-freeness with his Tree Adjoining Grammar. Even Chomsky's updated versions of transformational grammar [13], [14], [15] had to constrain the power of his generative grammar by imposing filters in the forms of hard constraints. This is to enrich context-free phrase structure grammar. The issue did not, however, come to a

halt. Head-Driven Phrase Structure Grammar [16] and Lexical Functional Grammar [17] are also of a sort of similar nature.

This shows that the question of what natural language belongs to as far as the class of formal languages is concerned is still a moot point. Is it because we classify languages (both artificial and natural) in terms of the Chomskyan hierarchy? Or is it because of some sort of natural order that has been imposed upon natural languages such that categorizing them into an exact class is not possible within the boundaries of classes of language that we know of. It seems that both questions are related to each other in that the available classes of languages in Chomskyan hierarchy do not take us much further in identifying and thereby understanding what computational property natural language universally possesses. The problem with such an approach is that it is not still clear what it would mean to revise and modify Chomskyan hierarchy in view of the fact that we cannot determine the exact computational dimension all natural languages have. In sum, what is clear enough is that this dimension which can be called the *N*-th dimension of language has not been broadly explored. People are debating whether natural language is context-free or context-sensitive; nobody seems to be concerned about whether natural language has some unknown computational dimension separate, segregated from and independent of context-freeness and context-sensitivity. Nobody wonders whether it is reasonable at all to get mired into the possibility of natural language being context-free or context-sensitive, because right wisdom among the computational linguistics community says language is not context-free at all. So one possibility may be to explore how much and to what extent natural language is context-free [18]. Nobody has yet done it, but its implications may be nontrivial in some respects. Let's now turn to some examples to get a flavor of the complication that the hidden dimension of language creates. If we have cases like the following in natural language,

(1) The girl the cat the dog chased bit saw ... the boy.
(2) I know he believes she knows John knows I believe .... we are here.

we have a class of languages like $a^n b^n$ which is exactly the class of context-free languages. In (1) we have a rule of the form $(NP)^n (VP)^n$ and $(NP\ VP)^n$ in (2). But it is quite clear that such cases are just a fragment of the set of *all* possible natural language expressions. Context-sensitivity also appears in a range of natural language expressions (especially in Dutch). Consider the following from English,

(3) What did I claim to know without seeing?
(4) What did you believe without seeing?

Cases like these show multiple dependencies between 'what' and the gaps after 'know' and 'seeing' in (3), and between 'what' and the gaps after 'believe' and 'seeing'. They may well fall into a pattern of context-sensitivity. Even such cases are pretty clear as to the way they reveal computational properties of context-sensitivity. What is not clear is that these examples neither explicitly demonstrate whether this can suffice for us to characterize natural language invariably as context-free or context-sensitive as long as one looks at it holistically, nor do they show the exact computational property within which both context-freeness and context-sensitivity

can be accommodated. Overall, it seems that in showing different fragments of natural language expressions from different languages as context-free and context-sensitive, we are missing out some fundamental generalizations which might underlie the way natural language shows both context-sensitive and context-free properties. We still do not know what these generalizations may amount to, but they clearly point to the vacuum as it is unclear how to fit the property of context-freeness into that of context-sensitivity.

All this makes it quite evident that natural language is neither fully context-free nor fully context-sensitive. If this is the case, then it is of some unknown computational property which is intermediate between context-freeness and context-sensitivity; and it may well be possible it has some property of type 0 grammar as well. Nobody knows. This is what we may term as the *N*-th dimension of language. Let's now move over to its nature.

3 The Nature of the *N*-th Dimension

A little bit of precision is now necessary. Hence, for the sake of precision, we will model this *N*-th dimension as a 4-tuple

$$\mathcal{D} = < D_n, E_p, \mathcal{P}_w, \infty >. \quad (1)$$

D_n is the set of hidden sub-dimensions characterizing and representing the components of the *N*-th dimension. E_p is the unknown expressive power of the *N*-th dimensional computational property of natural language. $\mathcal{P}_w$ is the Cartesian product of $D^{l}_{n} \otimes, ..., \otimes D^{m}_{n}$ where $m \neq n$. And ∞ is the distribution of the other three in a linear (or real or complex) space $\acute{S}$.

What is required at this stage is the stipulation that D_n determines how the unknown computational property appears whenever natural language is put under analysis, since this set contains the hidden components constituting the *N*-th dimension. So $D_n \vDash (\forall x)\, \lambda y\, [x \triangleq y]$ when $x \in D_n$ and $y \in P(D_n)^c$, the complement of the power set of D_n. This makes sure that it may well be possible that some components of the hidden *N*-th dimension of natural language correspond to some other components in the other class of computational properties in the Chomskyan hierarchy.

This is what leads us to make such claims that natural language is context-free or context-sensitive. No moving off, let's now say something about E_p. It can describe and generate all possible natural languages and all possible natural language expressions with its absolute generative capacity. Let's first denote $\mathcal{L}$ the set of all possible languages (both artificial and natural). And then let $\mathcal{L}^c$ denote the set of only and all possible natural languages. And $\boldsymbol{E}$ is the set of all possible expressions in natural language. So we can now say that

$$E_p \gtrdot \boldsymbol{E}(\mathcal{L}^c) \mid \acute{E} \subseteq \boldsymbol{E} \wedge \acute{E} = \phi \text{ when } E_p \not\gtrdot \acute{E}. \quad (2)$$

Here read the sign $\gtrsim$ as 'generable from'. This guarantees that there cannot exist any subset of the set of all possible expressions of natural languages which is not generable from E_p. At last ∞ orders the 4-tuple in configurations we are still unaware of. Roughly for the purpose of modeling, let's say there can possibly be $\mathbf{C}= \{C_1 , \ldots , C_k\}$ such configurations where k can be any arbitrary number. So we have

$$O_1(D_n), O_2(E_p), O_3(\mathcal{P}_w) \vdash O_i(\infty). \tag{3}$$

This is what expresses the ordering O_i of all the other three in the 4-tuple with respect to ∞. Now it becomes clear that the *N*-th dimension of language is quite complex in nature. Even if it can be mathematically modeled, it is at best patchy. We can never be sure that we have grasped it. For the sake of simplification, it was vital to model it. And now a bit of focus will be shed upon how it might work.

3.1 How the *N*-th Dimension Possibly Works

The description of how the *N*-th Dimension can possibly work is provided to facilitate an understanding of the way it works in natural language in general. In addition, this will also help us in proceeding one step ahead in realizing why it is so elusive as well that it has cudgeled so many brains of computer scientists and linguists both. Given that D_n is the set of hidden sub-dimensions characterizing and representing the components of the *N*-th dimension, we are now able to say that

$$\int f_1, \ldots, f_m(x_1 , \ldots, x_m)\, d\, x_1, \ldots , d\, x_m.\ \Delta\infty. \tag{4}$$

where $x_1 , \ldots, x_m \in D_n$ when $m \leq n$ such that $|D_n| = n$. This shows that $x_1 , \ldots, x_m \in D_n$ may appear in different distributions which determine differential probabilities associated with each distribution. The nature of such very differential probabilities themselves space out the way the *N*-th dimension appears in our analysis. Hence it is not simply a product of a linear function mapping a class of computational property into something, for example, from context-freeness into the *N*-th dimension. Using type predicate calculus [19], we can say that the following

$$\frac{\Gamma \vdash t_1 : T_1 , \ldots , \Gamma \vdash t_n : T_n}{\Gamma \vdash F(t_1, \ldots, t_n) : O_i(T_1, \ldots, T_n)\,.} \tag{5}$$

does not hold for the *N*-th dimension of natural language when Γ is a context, $t1$, ..., tn are computational properties of formal languages in the Chomskyan hierarchy like finite-state*ness*, context-freeness, context-sensitivity etc. which can be labeled as being of different types $T1$, ..., Tn . Let's us suppose here that the operator O_i fixes how the *N*-th dimension arises out of such a mapping from those computational properties $t1$, ..., tn into the types $T1$, ..., Tn. But even this does not capture the *N*-th

dimension as analyzing natural language as being a complex of different computational types does not help us much in that we do not know how O_i really works. We have just a vague idea of how it may function given that natural language has the properties of the types *T*1, ..., *Tn*.

Other dimensions of natural language also do not take us any farther in such a situation. We now know that a language has a grammar along with a lexicon and it generates expressions of that language. This is one of the (computational) properties of (natural) language. Given a grammar and a lexicon, the generated expressions of the grammar in that language are string sets. Of course language users do not think of their languages as string sets. However, this is another dimension. So is infinite recursion of string sets in natural language. Constraints on natural language constructions are determined by a set of universal principles. Pendar [20] has emphasized that natural language is a modularized computational system of interaction and satisfaction of such soft constraints. Optimality theory [21] a theory of natural language has built into it just such a system. These are some of the dimensions of language. Now consider the question: what if we add them together and then derive the *N*-th dimension since above all cumulative summing over all dimensions is equivalent to a higher dimension? A little bit of common sense shows that this may well be wrong. Let's see how. Let's sum all the existing dimensions of natural language

$$S^{(d)} = \sum_{i=1}^{n} d_i \qquad (6)$$

Here $S^{(d)}$ is never equivalent to the *N*-th dimension. Had it been the case, then by principle we would obtain the following

$$S^{(d)} = < D_n, E_p, \mathcal{P}_w, \infty >. \qquad (7)$$

But a careful consideration of the matter suggests that we do not get E_p out of $S^{(d)}$ even if the equivalence shows that we should. We do not get E_p out of $S^{(d)}$ mainly because $S^{(d)}$ may well generate some artificial language which does not have E_p. Analogically, let's suppose that entities in the world have only two dimensions, namely, width and length, but not height. In such a case combining the known will perhaps never give us the third one, height.

All this indicates that the *N*-the dimension is much more complex than has been recognized. However, it does not give us any reason for rejoicing in such a nature of natural language. Rather, it challenges us to decode its complexity which is so overwhelming that it is at present out of the bounds of our rational hold. This brings us to the next issue, that is, the relation of complexity to the *N-th* dimension.

3.2 Complexity and *N*-th Dimension

It may now be necessary to draw up connections between the *N*-th dimension and complexity. Is there any connection between them at all? How are they related? Of course, it is true that answers to such questions are not easy to get. We shall not try to answer those questions over here. Instead, an attempt will be made to map out the space over which the *N*-th dimension can be said to be complex. Now let's proceed. The notion of Kolmogorov complexity will be used here as it is handy enough in that it has been proved that it is machine independent [22]. We know that Kolmogorov complexity is the shortest possible program for a Universal Turing machine to compute the description of an object. Now let's assume that the object under consideration is the *N*-th dimension. What would its measure of complexity be with respect to Kolmogorov complexity? Mathematically speaking, we have

$$\mathcal{K}_u(\mathcal{D}) = \min_{p : u(p) = \mathcal{D}} l(p) \tag{8}$$

when $\mathcal{K}_u(\mathcal{D})$ is the Kolmogorov complexity computed by a Universal Turing machine u and p is the shortest possible program that can give a description of $\mathcal{D}$, the *N*-th dimension. If according to Kolmogorov complexity, the complexity of $\mathcal{D}$ is the length of the shortest possible program for computing $\mathcal{D}$, then such a program p does not independently exist. The term 'shortest possible program' is relative. Unless we find out a set of such programs P= $\{p_i, \ldots, p_k\}$, we can never measure the complexity of $\mathcal{D}$. But obviously we are sure that the $\mathcal{K}_u(\mathcal{D})$ is lower than l(this paper itself)! Now suppose the 4-tuple model that has been provided above to model $\mathcal{D}$ is actually equal to $\mathcal{K}_u(\mathcal{D})$. What is the implication then? So the length of any program that can model the 4-tuple is the complexity of ($\mathcal{D}$). Are we done with it? Possibly no. We still do not know whether this is *the* shortest program in all possible worlds. But if the program above is by the present criteria the complexity measure for $\mathcal{D}$, we have to accept that. However, the problem is still open ended, because we have not yet fully understood $\mathcal{D}$ at all despite the progress that has been made in formal language theory. This leads us to give the following definition

Definition (Unboundedness of the Complexity of $\mathcal{K}_u(\mathcal{D})$): Given that there may well exist a set of programs P= $\{p_i, \ldots, p_k\}$ whose actual length we are yet to figure out, $\mathcal{K}_u(\mathcal{D})$ is not bound from above by any constant C.

This definition ensures that we are still in uncertainty as to what the actual complexity of $\mathcal{D}$ can be. Science consists in simplification of complex phenomena. Here the *N*-th dimension of natural language is such a case. Even if we aim at simplifying it, it defies such attempts at simplification. That is what this section teaches us. This may be because of another important property of natural language, it is an emergent entity. Let's see what it has for the *N*-the dimension of natural language.

3.3 Emergence and the *N*-Dimension of Language

Now we have come near to the last candidate that has any claim that it can be related to the *N*-th dimension. It is not, of course, necessary that a link has to be forged between the *N*-th dimension and the case of emergence according to which the whole is not a sum of its parts. The case of natural language being emergent has been made quite forcefully by Andreewsky [23]. It appears that language is recursively emergent in that language emerges out of the interaction with several cognitive domains involving emotion, social cognition, vision, memory, attention, motor system, auditory system etc. at the higher level. And then at the linguistic level spanning out at the lower level, language emerges through integrated and interwoven, but partally constrained, interactions between syntax, semantics, morphology, lexicon and phonology which form an overlapping network. One type of emergence is thus embedded into another. The cognitive architecture of language at the higher level has been shown below

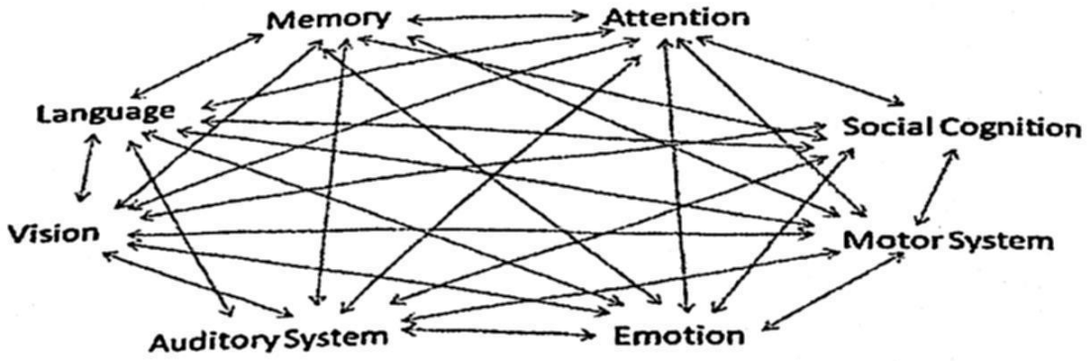

Fig. 1. Interconnection of language to other cognitive domains

Fig.1. above is symmetric with certain constraints on mutual interaction of the cognitive domains/systems such that not all information passes from one domain/system to another. Such constraints- computational, algorithmic and implementational in Marr's [24] sense- are specified by our genome. Here all these domains have varying interactions with respect to each other; what this means is that all these cognitive domains have differential mutual interactions with respect to each other. And by following Mondal [25] this can be represented through a distribution of joint probability in the following equation The interaction potential I(φ) can be characterized as

$$I(\varphi) = \sum_{i=1}^{N} P_i \int d_{c1,\ldots} d_{cN}\, \delta(_{c1 \ldots cN}).\ \Delta\psi. \qquad (9)$$

Here *c1* ... *cN* are differential probability functions coming out of the interaction dynamics of language with other cognitive domains and N in *c1* ... *cN* must be a finite arbitrary number as required for the totality of cognitive domains; *P* is a probability function; $\Delta\psi$ is a temporal distribution. The cognitive domains are functionally coherent units of emergent self-organizing representational resources realized in diffuse, often shared and overlapping, networks of brain regions as mediated by the bidirectional interactions in Fig. 1.

Now it may be asked what connection there exists between such cognitive emergence and the *N*-th dimension. Note that what is emergent is so complex that its complexity can be neither represented nor predicted. The above discussion on complexity and the *N*-th dimension just shows this much more clearly. In addition, emergent entities are chaotic and random at most times. Natural language is no exception. The way the lack of grasp of *N*-th dimension has been the centerpiece of debate on the computational property of natural language evidences the fact that it has appeared randomly as different to different researchers. That is why it has been once claimed to be of finite-state nature or sometimes context-free; sometimes mildly context-sensitive and so on. This is perhaps because of the emergent randomness and an apparent lack of order in the dimensions of natural language. One may also argue that here two separate issues- the computational property of linguistic structures and the evolution of language – are being conflated. But one should also note that whatever the (computational) properties of natural language are, they have come only through the mechanisms of evolution. Even many distinguishing properties of natural language like ambiguity, vagueness, redundancy, arbitrariness have appeared in natural language only as products or bi-products of evolution which is generally blind and so not goal-driven [26]. And it may well be plausible that it is such properties that confound and mystify the exact computational property of natural language. However, whatever the *N*-th dimension turns out to be, it will certainly be one of the greatest challenges for natural language processing, mathematical linguistics and theoretical computer science in general. This brings us nearer to the issue of what *N*-th dimension means for natural language processing.

4 What the *N*-th Dimension Means for Natural Language Processing

Grammar formalism is at the heart of natural language processing. Whether we are doing parsing, natural language generation, natural language understanding or machine translation etc., grammar formalisms are inevitable. Even if in recent years grammar induction through statistical techniques has been possible, the combination of robustness and minimum computational costs in terms of time and storage, still a goal aimed at, demands that a grammar formalism be available at hand. Even in cases where grammar induction is possible through machine learning, having a grammar formalism is always indispensable in that before the training period a standard model must be available for modeling [27]. In fact, there can be, in a more general sense, no parsing without a grammar. And parsing is at the heart of a range of natural language

processing tasks ranging from natural language generation, natural language understanding or machine translation to dialog systems etc. All present types of parsing use one or another grammar formalism based on different types of formal grammar [28], [29], [30], [31], [32], [33], [34].

So let's us confront the question of the *N*-th dimension in a rather straightforward manner. If we become successful some day in future in finding out this dimension, however chaotic and random it may well be, it will be one of the greatest achievements in the entire field of natural language processing and AI in general; for this will place us on a pedestal in building grammar models that can be implemented across languages universally. There will be no variation, no fractionation in the way grammar formalisms and models are built and implemented computationally. Once we coast to the *N*-th dimension of natural language processing, it will enable us in finessing and enhancing the effectiveness of our natural language processing tools which are still far behind compared to humans [35].

This will also help us in having a uniformity in our grammar models thereby eliminating the barrier of cross-linguistic transfer. This will propagate into multiple channels of progress in natural language technology in general. Just as the discovery of formal language hierarchy opened up new vistas for computation in general which ultimately gave birth to the field of natural language processing or NLP, the discovery of the *N*-th dimension will take us one revolutionary step farther toward a 'golden age' of natural language technology. One may complain that too much optimism is misplaced; but consider the sort of achievement in formal language theory this will amount to, as formal language theory has long been embroiled in the debates on the exact computational nature of natural language. After a lot of years, the efforts have been futile in that nobody has yet discovered the *N*-th dimension. What if we discover it some day? Would not it constitute a bigger achievement than even the discovery of Chomskyan hierarchy? Well, only the future will tell us. We are then nearing the conclusion with this ray of hope.

5 Concluding Remarks

This paper has sketched a brief sketch of what the *N*-th dimension is, its nature, its possible functioning and its links to complexity, emergence and NLP. After this brief tour it seems that we have achieved very little about the computational nature of natural language, even if we may know a lot about formal or artificial languages. The reason behind this is perhaps the *N*-th dimension itself. It is a major bottleneck in our path towards being able to compute all natural languages on earth. It may sound too ambitious, but any NLP researcher may be asked whether he/she does not want to see how his/her algorithms fare in other languages. The answer must at better be yes.

Moreover, it is also a dream for formal language theorists and perhaps mathematical linguists. In such a situation there is still enough scope for the claim that we may never be able to discover the *N*-th dimension of natural language because it is emergent, random and chaotic. We think such claims are as plausible as the claim that there does exist the *N*-th dimension of natural language to be discovered. Nobody can predict the path of the trajectory that research takes on in any field. The same applies

to NLP as well. Even if ultimately it transpires that the entire research program following on from the search for the computational class of natural language is flawed and wrong, this will still be fruitful in showing us the right path and a gateway towards a greater understanding of natural language.

References

1. Chomsky, N.: Syntactic Structures. Mouton, The Hague (1957).
2. Chomsky, N.: Formal Properties of Grammars. In: Luce, R. D., Bush, R. R., Galanter, E. (eds.) Handbook of Mathematical Psychology. Vol. II. Wiley, New York (1963).
3. Postal, P. M.: Limitations of Phrase-Structure Grammars. In: Fodor, Jerry, Katz, Jerald. (eds.) The Structure of Language: Readings in the Philosophy of Language. Prentice Hall, New Jersey (1964).
4. Langendoen, D. T.: On the Inadequacy of Type-2 and Type-3 Grammars for Human Languages. In: Hopper, P. J. (ed.) Studies in Descriptive and Historical Linguistics. John Benjamins, Amsterdam (1977).
5. Higginbotham, J.: English is not a Context-Free Language. Linguistic Inquiry. Vol. 15, 225-234 (1984).
6. Langendoen, D. T., Postal, P. M.: The Vastness of Natural Language. Blackwell, Oxford (1984).
7. Langendoen, D. T., Postal, P. M.: English and the Class of Context-Free Languages. Computational Linguistics. Vol. 10, 177-181 (1985).
8. Shieber, S. M.: Evidence against Context-Freeness of Natural Language. Linguistics and Philosophy. Vol. 8, 333-343 (1985).
9. Pullum, G. K., Gazdar, G.: Natural Languages and Context-Free Languages. Linguistics and Philosophy. Vol 4, 471-504 (1982).
10. Pullum, G. K.: On Two Recent Attempts to Show that English is not a CFL. Computational Linguistics. Vol. 10, 182-186 (1985).
11. Joshi, A.: Factoring Recursion and Dependencies: An Aspect of Tree-Adjoining Grammars and a Formal Comparison of some Formal Properties of TAGs, GPSGs, PLGs, and LFGs. In: Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics, pp. 7-15. Association for Computational Linguistics, Cambridge, MA (1983).
12. Joshi, A.: Tree Adjoining Grammars: How much Context-Sensitivity is Required to Provide Reasonable Structural Descriptions? In: Dowty, D., Karttunen, L., Zwicky, A. M. (eds.) Natural Language Processing: Psycholinguistic, Computational and Theoretical Perspectives. Cambridge University Press, New York (1985).
13. Chomsky, N.: Aspects of the Theory of Syntax. MIT Press, Cambridge, Mass. (1965).
14. Chomsky, N.: Rules and Representations. Columbia University Press, New York (1980).
15. Chomsky, N.: The Minimalist Program. MIT Press, Cambridge, Mass. (1995).
16. Pollard, C., Sag, I.: Head-Driven Phrase Structure Grammar. Chicago University Press, Chicago (1994).
17. Bresnan, J.: The Mental Representation of Grammatical Relations. MIT Press, Cambridge, Mass. (1982).
18. Pullum, G. K.: Systematicity and Natural Language Syntax. Croatian Journal of Philosophy. Vol. 7. 375-402 (2007).
19. Turner, R.: Computable Models. Springer, London (2009).
20. Pendar, N.: . Linguistic Constraint Systems as General Soft Constraint Satisfaction. Research on Language and Computation. Vol. 6:163–203 (2008).
21. Prince, A., Smolensky, P.: Optimality Theory: Constraint Interaction in Generative Grammar. Rutgers University Centre for Cognitive Science, New Jersey (1993).

22. Cover, T. M., Thomas, J. A.: Elements of Information Theory. Wiley, New York (1991).
23. Andreewsky, E.: Complexity of the Basic Unit of Language: Some Parallels in Physics and Biology. In: Mugur-Schachetr, M., van der Merwe, A. (eds.) Quantum Mechanics, Mathematics, Cognition and Action. Springer, Amsterdam (2002).
24. Marr, D.: Vision: A Computational Investigation into the Human Representation and Processing of Visual Information. W H Freeman, San Francisco (1982).
25. Mondal, P.: How Limited is the Limit? In: Proceedings of the International Conference on Recent Advances in Natural Languages Processing, pp. 270-274. RANLP, Borovets (2009).
26. Kinsella, A. R., Marcus, G. F.: Evolution, Perfection and Theories of Language. Biolinguistics. Vol 3.2–3: 186–212 (2009).
27. Ristad, E. S.: The Language Complexity Game. MIT Press, Cambridge, Mass. (1993).
28. Goodman, J.: Semiring Parsing. Computational Linguistics. Vol. 25, 573.605 (1999).
29. Charnaik, E.: A Maximum-Entropy-Inspired Parser. In: Proceedings of North American Chapter of the Association for Computational Linguistics-Human Language Technology, pp. 132-139. Association for Computational Linguistics, Washington (2000).
30. Eisner, J.: Bilexical Grammars and their Cubic-Time Parsing Algorithms. In: Bunt, H., Nijholt, A. (eds.) Advances in Probabilistic and Other Parsing Technologies, pp. 29.62. Kluwer Academic Publishers, Amsterdam (2000).
31. Nederhof, M. J.: Weighted Deductive Parsing and Knuth's Algorithm. Computational Linguistics. Vol. 6. 135.143 (2003).
32. Pradhan, S., Ward, W., Hacioglu, K., Martin, J. H., Jurafsky, D.: Shallow Semantic Parsing Using Support Vector Machines. In: Proceedings of North American Chapter of the Association for Computational Linguistics-Human Language Technology, pp. 233-240. Association for Computational Linguistics, Boston, MA (2004).
33. Charnaik, E., Johnson, M.: Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, 173-180. Association for Computational Linguistics, Michigan (2005).
34. Huang, L., Chiang, D.: Better k-Best Parsing. In: Proceedings of International Workshop on Parsing Technologies (IWPT), pp. 53-64. Vancouver, Omnipress (2005).
35. Moore, R. K.: Towards a Unified Theory of Spoken Language Processing. In: Proceedings of 4th IEEE International Conference on Cognitive Informatics, pp. 167-172. IEEE Press, Irvine, CA (2005).

Automatic Derivational Morphology Contribution to Romanian Lexical Acquisition

Petic Mircea

Institute of Mathematics and Computer Science of the ASM,
5, Academies str., Chisinau, MD-2028, Republic of Moldova
mirsha@math.md

Abstract. The derivation with affixes is a method of vocabulary enrichment. The consciousness of the stages in the process of the lexicon enrichment by means of derivational morphology mechanisms will lead to the construction of the new derivatives automatic generator. Therefore the digital variant of the derivatives dictionary helps to overcome difficult situations in the process of the new word validation and the handling of the uncertain character of the affixes. In addition, the derivatives groups, the concrete consonant and vowel alternations and the lexical families can be established using the dictionary of derivatives.

1 Introduction

The linguistic resources represent the fundamental support for automatic tools development in the processing of linguistic information. The lexical acquisition constitutes one of the most important methods for lexical resources enrichment. The examination of the problems referring to automatization is one of the main aspects in the process of the linguistic resources creation.

The need of the lexical resources enrichment is satisfied not only by borrowings of words from other languages, but also by the use of some exclusively internal processes [1]. Inflection, derivation and compounding are the most useful ways of word formation in Romanian language. *Inflection* is the generation of word forms without changing their initial meaning. *Derivation* means the creation of a new word by adding some affixe(s) to existent lexical base. *Compounding* is made up of the words which exist and are independent or with the elements of thematic type. In this article the derivation is investigated.

The aim of this article is to study the derivational morphology mechanisms which permit the automatic lexical resources acquisition for Romanian language.

In this context the paper is structured in the following way. Firstly, the derivational particularities of Romanian language are described, and then the most important stages in the automatic derivation are the subject to review. A special description is dedicated to the methods of new word validation. The uncertain particularities of Romanian affixes are examined. The solution for the uncertainty was found in the digital variant of the Romanian derivatives dictionary. The new electronic resource can be used in the process of detection of the derivatives groups. Moreover, it permits the vowel and consonant alternations examination and the construction of lexical families in the derivation process.

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 67-78

Received 22/11/09
Accepted 16/01/10
Final version 09/03/10

2 Derivational Particularities of Romanian Language

The majority of Romanian language derivational mechanisms are also common for other European languages, especially for those of Latin origin.

The *affix* represents any morpheme which remains beside the root, when a word is segmented. It includes *prefixes* (added before the root) and *suffixes* (added after the root) [2]. Romanian language has 86 prefixes [3] and more than 600 suffixes. There are two types of suffixes: lexical and grammatical. *Lexical suffix* is a group of letters (or a single letter) which is added after the root and forms a new word. *Grammatical suffix* is a group of letters (or a single letter) which is added after the stem and, as a result, a form of the same word is obtained [4]. Roots, prefixes and suffixes represent the *morphemes* [5].

The word formed by adding a prefix or a suffix is called *derivative*. For example, the derivative *frumuseţe*, consists of the root *frumos* and the suffix *-eţe*. Often suffixes and prefixes are not added directly to roots but to a lexical stem, which represents a root and, at least, a prefix or a suffix, for example, *străbătător* is formed from the prefix *stră-* and the stem *bătător*, which consists of the root *bate* and the suffix *-ător* [2].

The derivatives can be classified in three groups: analyzable, semi-analyzable, and non-analyzable. In the *analyzable* derivatives both the affix and the root are distinguished. *Semi-analyzable* derivatives represent the words in which only the affix is distinguished. In the *non-analyzable* derivatives we can not distinguish neither the affix nor the root [3].

3 Stages in the Automatic Derivation

In the process of the derivational morphology automatization it is important to examine the following steps: the analysis of the affixes, the elaboration of affixes detection algorithm, the formalization of the derivational rules and the validation of the new words generated by specific algorithms.

The analysis of the affixes consists in the establishing of its quantitative features. The the quantitative features for some prefixes and suffixes were set up according to the list of Romanian prefixes and suffixes. On the base of the elaborated algorithms [6], programs were developed that allowed to find out:

- the number of words which begin (end) with some prefixes (suffixes);
- the number of words derived from these affixes;
- the repartition of letters followed by the mentioned affixes;
- the part of speech distribution for some Romanian affixes.

Taking into account the obtained results, it was noticed that some prefixes and suffixes have more phonological forms. That is why it is difficult to set up the quantitative characteristics of the affixes [6].

As not all the words end (begin) with the same suffixes (prefixes), some algorithms were elaborated for enabling the automatic extraction of the derivatives from the lexicon . The elaborated algorithms took into account the fact that being $x, y \in \Sigma^+$, where Σ^+ is the set of all possible roots, and if $y = xv$ then v

is the suffix of y and if $y = ux$ then u is the prefix of y. In this context both y and x must be valid words in Romanian language, and u and v are strings that can be affixes attested for Romanian language [3]. The problem of consonant and vocalic alternations was neglected in the case of the algorithm derivatives extraction. This fact does not permit the exact detecting of all derivatives.

For a certain number of affixes was found out some derivational rules which allow the generation of new derived words unregistered in the dictionaries [7]. In the process of the completion of linguistic resources by automatic derivation appears a natural tendency of using the most frequent affixes. But, in fact, the use of most productive affixes become to be problematic because of their irregular behavior [7]. That is why those affixes which permit to formulate simpler rules of behavior without having many exceptions were taken for the research. As a consequence, these rules operate with prefixes *ne-* and *re-*, and also with suffixes *-tor* and *-bil* (the last being frequent in the derivational process with the prefix *ne-*). The lexical suffix *-iza* was also included in the research, having neological origin and being actually very productive in the strong relation with the lexical suffixes *-ism* and *-ist*.

After derivatives generation process, not all obtained words can be considered valid. The set of words needs to pass the step of validation, which represents an important level in the correct detection of the generated derivatives, thanks to the programs based on derivation rules. Also, it is possible to validate the words with the help of linguists, but it requires more time and there is the possibility of making mistakes.

On the other hand, validation can be done by checking the derived words presence in the electronic documents. In this situation it is possible to use verified electronic documents, such as different Romanian corpora. Unfortunately, the corpora have insufficient number of words, which can be used in new derivatives validation. Therefore, the simplest way is to work with the existent documents on Internet, which are considered to be unverified sources. In this case the difficulties appear in setting up the conditions which assure a valid word. So, it was attempted to determine the indices of frequency [7] for some Romanian affixes. Thus, a program which extracts the number of appearances of words in Google searching engine for Romanian language was elaborated.

4 Digital Variant of the Derivatives Dictionary

The derivatives dictionary [8] has only the graphic representation of the derivative and the constituent morphemes without any information about the part of speech of the derivatives and of its stems. The digital variant of the dictionary [8] is obtained after the process of scanning, OCR-izing and the correction of the original entries. This electronic variant of the dictionary [8] becomes important as it is difficult to establish the criterion for validation of new derived words. Nevertheless, it permits the detecting of the derivatives morphemes with its type (prefix, root and suffix) and constitutes an important electronic linguistic resource.

Practically, the inputs in this dictionary are constructed being based on an unsure scheme. It is not clear where the affixes and the root are. In order to exclude the uncertainty in the input of the digital variant of the dictionary, it was made a regular expression that represents the structure of the derivatives:

$$derivative = (+morpheme)^*.morpheme(-morpheme)^*$$

where $+morpheme$ is the prefix, $.morpheme$ is the root and $-morpheme$ is the suffix.

Table 1. The most frequent prefixes

Prefix	Number of distinct derivatives
ne-	571
în-	293
re-	281
des-	109
pre-	109

To find out the statistic characteristics of the dictionary algorithms were elaborated and then programs were developed. It was calculated that dictionary consists of 15.300 derivatives with 42 prefixes (Table 1), 433 suffixes (Table 2) and over 6800 roots.

Table 2. The most frequent suffixes

Suffix	Number of distinct derivatives
-re	2793
-tor	605
-toare	522
-ală	514
-ie	400

5 Uncertain Characters of the Affixes

One of the main problems concerning Romanian derivation is the uncertainty of morpheme boundaries. In many cases different people, or even the same person put in different situations, divides the same word into segments in different ways. Besides the segmentation in morphemes (for example: anti-rachet) or in the allomorph variants, a word form can be segmented in different ways [8]

such as: *syllables* (for example: an-ti-ra-che-tă), *sounds* (for example: a-n-t-i-r-a-ch-e-t-ă) and *letters* (for example: a-n-t-i-r-a-c-h-e-t-ă). All these types of segmentation have nothing to do with segmentation in morphemes, because they do not indicate the morpheme boundaries.

The segmentation in morphemes implies the detection of the morphemes with its types (root, prefix, and suffix). Unfortunately, not all morphemes are different. It was observed that the derivatives can contain roots which can be also suffixes or prefixes (for example: *-re*, *-tor*, *-os*, *-uşor*, *-uliţă*). Also, there are morphemes where the root, prefix and suffix coincide, for example, the morpheme an is a prefix in the word *an*istoric, in the word *an*işor is a root and in the word americ*an* is a suffix.

Taking into account this uncertain character of the morphemes boundaries within the derivatives, it appears a natural tendency of applying to the probabilistic method in measuring this uncertainty.

Let X be a discrete variable with the values $\{x_1, x_2, x_3, x_4, x_5, x_6\}$, where
x_1 represents the case when a string is a prefix, for example, *an*istoric;
x_2 – a part of a prefix, for example, *an*tevorbitor;
x_3 – a root, for example, *an*işor;
x_4 – a part of a root, for example, um*an*;
x_5 – a suffix, for example, americ*an*;
x_6 – a part of a suffix, for example, comedi*an*t.

Table 3. Prefixes with the lowest entropy

String	Number of prefixes	Number of parts of prefixes	Number of roots	Number of parts of roots	Number of suffixes	Number of parts of suffixes	Entropy
super	6	0	0	0	0	0	0.0000
ultra	12	0	0	0	0	0	0.0000
arhe	1	0	0	0	0	0	0.0000
com	1	0	0	63	0	0	0.1161
auto	87	0	0	6	0	0	0.3451

Let $p(x_k)$ be the probability of the x_k event. Of course, not all the affixes can correspond to all the values of the discrete variable X. Therefore, let consider the set:

$$Y = \{y_k | p(x_k) \neq 0, 1 \leqslant k \leqslant 6\}.$$

Let $H(Y)$ be the entropy of the Y:

$$H(Y) = -\sum_{i=1}^{6} p(y_i) \times log_2 p(y_i)$$

For example, the prefix *răs-* corresponds to three values of the discrete variable X. So, the set is $Y = \{y_1, y_2, y_4\}$. The respective probabilities are $p(y_1) = 0,34313, p(y_2) = 0,01960, p(y_4) = 0,63725$.

$$H(Y) = -(p(y_1) \times log_2 p(y_1) + p(y_2) \times log_2 p(y_2) + p(y_4) \times log_2 p(y_4)) = 1,05495$$

It is worth being mentioned that if the uncertainty is higher, it means that the number of cases will increase, so the entropy will increase too, otherwise it will decrease. For example for the prefix *ultra-*, as a result, it is used only as a prefix, that is why the entropy is 0 (Table 3). In the case of the prefix *an-* the entropy is 1,2322 (Table 4), because it can correspond to all six values of the discrete variable X.

Table 4. Prefixes with the highest entropy

String	Number of prefixes	Number of parts of prefixes	Number of roots	Number of parts of roots	Number of suffixes	Number of parts of suffixes	Entropy
re	281	118	0	4708	3218	975	1.6005
intra	2	0	2	1	0	0	1.5219
an	1	105	1	1058	74	200	1.2322
de	71	209	0	547	0	0	1.2000
auto	571	0	0	426	0	39	1.1790

6 The Analysis of the Consonant and Vowel Alternations

The problem of derivation consists not only in the detection of the derivational rules for separate affixes, but also in the examination of the concrete consonant and vowel alternations for the affixes. It is important that not all affixes need vowel and consonant alternations in the process of derivation. On the purpose of precising which affixes have alternations in the process of derivation the digital variant of the derivatives dictionary was studied (Table 5).

The situation is that there are more derivatives without alternations, especially in the case of the prefix derivation. The lack of vowel and consonant alternations in the process of derivation is observed with the following most frequent prefixes: *ne-*, *re-*, *pre-*, *anti-*, *auto-*, *supra-*, and *de-*. The prefixes *în-*, *des-*, *sub-*, *dez-*, and *îm-* use the vowel and consonant alternations in the process of derivation (Table 6).

There are several types of vowel and consonant alternations in the process of derivation with the prefixes:

Table 5. Statistics about the derivatives

Affix	Number of derivatives witout alternations	Number of derivatives with alternations
prefix	1134	224
suffix	6809	6381
prefix and suffix	632	191
total	8575	6796

– the addition of a letter to the end of the root, for example, *şurub* → *înşuruba, bold* → *îmboldi, plin* → *împlini*;

– the changing of the final letter in the root, for example, *lînă* → *dezlîna, purpură* → *împurpura, puşcă* → *împuşca*;

– the changing of the final letter in the root and the addition of the letter to the end of the root, for example, *avut* → *înavuţi, compus* → *descompune, păros* → *împăroşa, blînd* → *îmblînzi*;

– the changing of the vowels in the root, for example, *cataramă* → *încătărăma, primăvară* → *desprimăvăra, rădăcină* → *dezrădăcina, platoşă* → *împlătoşi*;

– the changing in the prefix, for example, *şoca* → *deşoca, pat* → *supat*;

– the avoiding of the double consonant, for example, *spinteca* → *despinteca, braţ* → *subraţ*.

Table 6. Prefixes with vowel and consonant alternations

Prefix	Number of derivatives witout alternations	Number of derivatives with alternations	Total number of derivatives
în-	33	115	148
des-	57	6	63
sub-	81	2	83
dez-	32	2	34
îm-	3	38	41
total	206	163	369

In the most of cases in the process of derivation with prefixes *în-* and *îm-* the alternations are used because of the part of speech changing, especially from adjectives and nouns to verbs.

The process of derivation with suffixes does not attest cases without consonant and vowel alternations. It means that there are situations when the derivation is made up with minimum number of alternations (Table 7) and with maximum cases of changes in the root (Table 8). The possible vowel and consonant alternations are so varied that it is difficult to describe them all in a chapter, but it is possible, at least, to classify them:

Table 7. Suffixes with fewer derivatives formed without alternations

Suffix	Number of derivatives witout alternations	Number of derivatives with alternations	Total number of derivatives
-re	2782	11	2793
-tor	561	43	605
-toare	478	35	522
-iza	221	23	244
-tură	205	27	232

– the changing of the final letter in the root, for example, *alinia* → *aliniere*, *aşchia* → *aşchietor*, *cumpăra* → *cumpărător*, *curăţi* → *curăţător*, *delăsa* → *delăsător*, *depune* → *depunător*, *faianţă* → *faianţator*, *fărîma* → *fărîmător*, *împinge* → *împingător*, *transcrie* → *transcriitor*;

– the removing of the last vowel in the root, for example, *răşchia* → *răşchitor*, *acri* → *acreală*, *aduna* → *adunătoare*;

– the removing of the final vowel in the root and the changing of the letter before the last one, for example, *zeflemea* → *zeflemitor*, *ascunde* → *ascunzătoare*;

Table 8. Suffixes with fewer derivatives formed without alternation

Suffix	Number of derivatives witout alternations	Number of derivatives with alternations	Total number of derivatives
-eală	19	495	514
-ătoare	5	281	286
-ător	5	279	284
-ar	110	249	359
-ie	166	234	400

– the changing of two final letters in the root, for example, *bea* → *băutor*, *bea* → *băutoare*, *încăpea* → *încăpătoare*;

– the changing of the first letter of the suffix, for example, *bîntui* → *bîntuială*, *murui* → *muruială*;

– the removing of the final letter in the root with the vowel changing, for example, *cană* → *căneală*, *atrage* → *atrăgătoare*, *bate* → *bătătoare*;

– the removing of the two letters in the suffix, for example, *căpia* → *căpială*, *încleia* → *încleială*;

– the removing of the final letter and that of the vowel inside the root, for example, *coace* → *cocătoare*;

– the removing of the final vowel and the changing of the final consonant, for example, *descreşte* → *descrescătoare, închide* → *închizătoare, încrede* → *încrezătoare, promite* → *primiţătoare*;

– the changing in the root, for example, *rîde* → *rîzătoare, recunoaşte* → *recunoscătoare, roade* → *rozătoare, sta* → *stătătoare, şedea* → *şezătoare, vedea* → *văzătoare, şti* → *ştiutor.*

7 Lexical Families

The set of derivatives with a common root and meaning represents a lexical family. So, the second part of the definition is very important because there is a tendency of grouping the words in lexical families only by a common root. Therefore the following words: *alb*, *albastru*, and *albanez* can be considered lexical family, so ignoring the fact that lexical family consists of the words that have the appropriate meaning. In addition, a lexical family consists of the words which are different in what concern the grammatical categories, beginning with the same root. The word base can have the same form of the root for all words from the family, for example: *actor actoraş, actoricesc, actorie.*

When in a derivative the affixes are trimmed, it means that only the root remains. The root can suffer little alternations: *ţară* (*ţar-/ ţăr-*); but it is never changed. During the derivation, it was observed that not all the lexical units derive directly from the root, some of them derive from previous derivatives, for example,

$$[\text{cruce}]_{noun} \rightarrow [\text{cruciş}]_{adj,adv} \rightarrow [\text{încrucişa}]_{verb} \rightarrow [\text{încrucişator}]_{adj,noun}.$$

Lexical family consists also of compounds, which contain the word base from the respective families. So, the compound word *cal-de-mare* is in the lexical family of the word *cal* and also in that of the word *mare*. But, it will not be in the lexical family of the word *călare*, which is the word base of the other family: *călari, călarie, călarime, călaresc, călaraş.*

In the electronic variant of the derivatives dictionary the most numerous lexical families are of the root **bun** (32 derivatives with the prefixes: *stră-, îm-, ne- and în-*; and the suffixes: *-el, -eţe, -ătate, -ic, -uţă, -icea, -icel, -icică, -işoară, -işor, -iţă, -uţ, -re, -i, -ariţă, -atic, -eală, -ească, -esc, -eşte, -toare, -tor*, and *-ie*); **alb** (25 derivatives with the prefix: *în-*; and with the suffixes: *-eaţă, -ei, -eţ, -eală, -icioasă, -icios,-iliţă, -re, -i, -ime, -ineaţă, -ineţ, -ior, -işor, -itoare, -itor, -ie, -itură, -iţă, -ui, -uie, -uleţ, -uş*, and *-uţ*); **şarpe** (22 derivatives without any prefixes and with the suffixes: *-ar, -aş, -ărie, -ească, -esc, -eşte,-işor, -oaică, -oaie, -oi, -ui, -eală, -re, -toare, -tor, -tură, -urel*, and *-uşor*); **roată** (22 derivatives without any prefixes and with the suffixes: *-ar, -easă, -ie, -it, -iţă, -aş,-at, -ată, -i, -cică, -re, -tor, -toare, -tură, -ilă, -at, -iş, -ocoală*, and *-ocol*); **om** (20 derivatives with the prefixes: *ne-* and *supra-*; and with the suffixes: *-ime, -oasă, -os, -oi, -uleţ, -uşor, -ească, -esc, -eşte*, and *-ie*). In the same context there are over 3000 roots with a single derivative.

There were found out that 7 prefixes and namely *a-*, *arhe-*, *para-*, *dis-*, *i-*, *im* and *întru-*, are not attached directly to roots, but only to stems. Also there are several suffixes, that are not attached to root.

8 The Process of Derivative Groups Establishing

8.1 Derivative Groups with Prefixes

The derivatives were extracted separately for every affix and were compared with the flection groups. Thanks to flection groups from [9, 10] and derivatives from morphological dictionary [8] it was attempted to detect the derivation groups. The morphological dictionary [8] consists already of many derivatives.

So, in order to acquire information referring to affixes, which can be attached to roots from the dictionary [9] special programs were developed. The derivatives those which have the following prefixes proved to be the most numerous, in a descending order: *ne-*, *re-*, *în-*, *des-*, *pre-*, *anti-*, *auto-*, *sub-*, *dez-*, *supra-*, *de-*, and *îm-*. The rest of the prefixes have an unsignificant number of derivatives. These 12 prefixes from 42 form 88.2 per cent from all derivatives with prefixes, registered in this electronic dictionary.

Firstly, it was set up the flection groups of the roots, which correspond to derivatives with prefixes without any suffixes. Nevertheless, there is a big number of flection groups for a single prefix, for example,

$$\texttt{nescris=+ne.scris}_{N29}$$
$$\texttt{nescris=+ne.scris}_{N1}$$
$$\texttt{nescris=+ne.scris}_{N24}$$
$$\texttt{nescris=+ne.scris}_{A4}$$
$$\texttt{nescris=+ne.scris}_{M6}.$$

For every prefix was set up the most frequent flection group of the derivatives roots. So, the roots with verbal flection group V201 is attached mostly to the following prefixes: *re-*, *des-*, *auto-*, *dez-*, *de-*; masculine noun flection group M1: *anti-*, *sub-*, *îm-*; feminine noun flection group F1: *în-*, *supra-*; verbal flection group V401: *pre-*; adjectival flection group A2: *ne-*.

Secondly, it was extracted the flection group of the roots that correspond to derivatives with prefixes that was first derivated with suffixes. In this case, there is the probability that there are roots with the same flection groups that form derivatives with different suffixes, for example,

$$\texttt{subordonator=+sub.ordona}_{V201}\texttt{-tor}$$
$$\texttt{subordonatoare=+sub.ordona}_{V201}\texttt{-toare}.$$

In the same procedure as that described above the derivatives are used with the most frequent flection groups of the derivatives roots: verbal flection group V201 uses mostly the following prefixes: *pre-*, *dez-*, *auto-*, *sub-*, *de-*, *supra-*; verbal flection group V401: *ne-*, *des-*; masculine noun flection group M1: *îm-*; adjectival flection group A1: *anti-*; neutral noun flection group N24: *în-*.

During the investigation there were found out that the following prefixes are attached especially to the nouns: *ne-, în-, anti-, sub-, supra* and *în-*. Another group of prefixes is in the most of cases attached to the verbs: *de-, dez-, auto-, pre-, des-* and *re-*.

8.2 Derivative Groups with Suffixes

In the same way as for the prefixes, in order to decide to which roots can be attached the concrete suffix from the morphological dictionary, special programs were developed, which extracted derivatives separately for every suffix, and after that, it is compared with the flection group.

The most numerous derivatives proved to be, in a descending order, the following suffixes: *-re, -tor, -toare, -eală, -ie, -ătoare, -iza, -oasă, -ar, -ător, -ească, -os, -aş, -esc, -tură, -iţă, -ist, -uţă, -el, -i, -ui, -ătură, -eşte, -ism, -a, -ărie, -ică, -ime, -itate, -ioară, -işor, -işoară, -ic, -uleţ, -că, -ean, -iş, -easă, -bil, -uţ, -at, -oaică, -uşor, -an, -oi, -uliţ, -iu, -enie, -istă, -al,* and *-ea*. The rest of the suffixes have an insignificant number of derivatives. From 430 suffixes registered in this electronic dictionary 52 of them form 87.7 per cent from suffix derivatives.

It was retrieved the flection group of the roots that correspond to derivatives with suffixes. The words in the dictionary [9] have several entries for different flection groups, for example, the verb *învîrti* takes part from verbal flection groups `V401` and `V305` (the same part of speech), and the word *croi* belongs to different parts of speech: noun flection group `N67` and verbal flection group `V408`.

For every suffix it was established the most frequent flection groups for the derivatives roots. So, from the roots with the masculine noun flection group `M1` flection group can be generated the derivatives with the following suffixes: *-ie, -ească, -aş, -esc, -iţă, -el, -i, -eşte, -ism, -ime, -ic, -iş, -oaică, -an, -oi, -iu, -uţ*; neutral noun flection group `N24`: *-oasă, -os, -arie, -işor, -uleţ, -ean, -uţ, -uşor, -al*; feminine noun flection group `F1`: *-ar, -uţă, -ică, -işoară, -uliţă*; verbal flection group `V401`: *-tor, -toare, -eală, -tora, -enie*; verbal flection group `V201`: *-re, -ătoare, -ător, -ătură, -bil*, `N1`: *-ist, -at, -al* (also neutral noun flection group `N24`), adjectival flection group `A1`: *-iza, -itate*, feminine noun flection group `F135`: *-ioară*, masculine noun flection group `M20`: *-că*, neutral noun flection group `N11`: *-istă*, feminine noun flection group `F43`: *-ea*.

After the examinations of the most frequent suffixes it was found that the following suffixes are attached mostly to the nouns: *-ie, -iza, -oasă, -ar, -ească, -os, -aş, -esc, -tură, -iţă, -ist, -uţă, -el, -i, -ui, -eşte, -ism, -a, -ărie, -ică, -ioară, -işor, -işoară, -ic, -uleţ, -că, -ean, -iş, -easă, -uş, -at, -oaică, -uşor, -an, -oi, -uliţă, -iu, -enie, -istă, -al, -ea*, and *uţ*. Another group of suffixes, and namely: *-re, -tor, -toare, -eală, -ătoare, -ător, -tură, -ătură*, and *-bil*, is attached mostly to verbs. Only two suffixes are attached in the most of cases to adjectives: *-ime* and *-itate*.

9 Conclusions

The process of the derivatives generator construction needs the detailed studying of the affixes and the derivatives features. The new derivatives validation is one of the steps in automatic derivation that raises many questions.

In the case it is difficult to set up the criterion for words validation by means of Internet, it is important to use the digital variant of the derivatives dictionary, which will permit the establishing of the morphemes of the derivatives with its type (prefix, root and suffix). It was proved to be useful in the detection of entire variety of consonant and vowel alternations in the process of derivation with prefixes and suffixes.

Another notion connected with the lexical derivation is the lexical family, that offers the possibility of acquiring sets of affixes possible to attach to the concrete roots.

The grouping of the derivatives by flection classes can give the possibility of finding the morphological characteristics of the words. The following step in the research should answer whether the most numerous flection groups for affixes can serve for the process of automatic generation of new valid derivatives.

References

1. Tufiş, D., Barbu, A.M., Revealing Translator's Knowledge: Statistical Methods in Constructing Practical Translation Lexicons for Language and Speech Processing, *International Journal of Speech Technology* 5, 2002, pp. 199-209.
2. Hristea, T., *Sinteze de limba română*, Bucureşti, 1984, pp. 66-99.
3. Graur, Al., Avram, M., *Formarea cuvintelor în limba română*, vol. II Editura Academiei, Bucureşti, 1978.
4. *Gramatica limbii române. Cuvîntul*, Editura Academiei Române, Bucureşti, 2005.
5. Hausser, R., *Foundations of Computational Linguistics. Human-Computer Communication in Natural Language*, 2nd Edition, Revised and Extended, Springer, 2001, 580 p.
6. Petic, M., Specific features in automatic processing of formations with prefixes, *Computer Science Journal of Moldova*, 4 1(7), 2008, pp. 209-222.
7. Cojocaru, S., Boian, E., Petic, M., Stages in automatic derivational morphology processing, *International Conference on Knowledge Engineering, Principles and Techniques, KEPT2009, Selected Papers*, Cluj-Napoca, July 24, 2009, pp. 97-104.
8. Constantinescu, S., *Dicţionar de cuvinte derivate. Editura Herra*, Bucureşti, 2008.
9. Lombard, A., Gâdei, C., *Dictionnaire morphologique de la langue roumaine*, Editura Academiei, Bucureşti, 1981, 232 p.
10. S.Cojocaru, M.Evstunin, V.Ufnarovski, Detecting and correcting spelling errors for the Roumanian language. *Computer Science Journal of Moldova*, vol.1, no.1(1), 1993, pp 3-21.

POS-tagging for Oral Texts with CRF and Category Decomposition

Isabelle Tellier[1], Iris Eshkol[2], Samer Taalab[1], and Jean-Philippe Prost[1,3]

[1] LIFO, Université d'Orléans, France
[2] LLL, Université d'Orléans, France
[3] INRIA Lille - Nord Europe, France
{name}.{lastname}@univ-orleans.fr

Abstract. The ESLO (Enquête sociolinguistique d'Orléans, *i.e. Sociolinguistic Survey of Orléans*) campaign gathered a large oral corpus, which was later transcribed into a text format. The purpose of this work is to assign morpho-syntactic labels to each unit of this corpus. To this end, we first studied the specificities of the labels required for oral data, and their various possible levels of description. This led to a new original hierarchical structure of labels. Then, since our new set of labels was different from any of those of existing taggers, which are usually not fit for oral data, we have built a new labelling tool using a Machine Learning approach. As a starting point, we used data labelled by Cordial and corrected by hand. We used CRF (Conditional Random Fields), to try to take the best possible advantage of the linguistic knowledge used to define the set of labels. We measure accuracy between 85 and 90, depending on the parameters.

1 Introduction

Morpho-syntactic tagging is essential to text analysis, as a preliminary step to any high level processing. Different reliable taggers exist for French, but they have been designed for handling written texts and are, therefore, not suited to the specificities of less "normalised" language. Here, we are interested in the ESLO[4] corpus, which comes from records of spoken language. ESLO thus presents specific features, which are not well accounted for by standard taggers.

Several options are possible to label transcribed spoken language: one can take a tagger initially developed for written texts, providing new formal rules let us adapt it to take into account disfluences (Dister, 2007 [1]); or one can adapt the transcribed corpus to the requirements of written language (Valli and Veronis, 1999 [2]). We have chosen a different methodological approach. Starting from the output of a tagger for written language, we have first defined a new tag set, which meets our needs; then, we have annotated a reference corpus with those new tags and used it to train a Machine Learning system.

For that kind of annotation task, the state-of-the-art technology for supervised example-based Machine Learning are the Conditional Random Fields

[4] Enquête sociolinguistique d'Orléans, *i.e. Sociolinguistic Survey of Orléans*

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 79-90

Received 27/11/09
Accepted 16/01/10
Final version 09/03/10

(CRF). CRF is a family of recently introduced statistical models (Lafferty *et al.*, 2001 [3], Sutton and McCallum, 2007 [4]), which have already proven their efficiency in many natural language engineering tasks (McCallum and Li, 2003 [5], Pinto *et al.*, 2003 [6], Altun *et al.*, 2003 [7], Sha and Pereira, 2003 [8]). Our experiments make use of CRF++[5], a free open-source library, developed by Taku Kado. We proceed with the testing of various strategies for decomposing the labels into a hierarchy of simpler sub-labels. The approach is original, in that it eases the learning process, while optimising the use of the linguistic knowledge that ruled the choice of intial labels. In that, we follow the procedure suggested by Jousse (2007 [9]), and Zidouni (2009 [10]).

In the following, Sect. 2 is dedicated to the presentation of our corpus and of the tagging process, focusing on the labelling problems raised by spoken language. After justifying the choice of a new tag set, we explain the procedure we have adopted for acquiring a sample set of correctly labelled data. In Sect. 3 we present the experiments we have carried out with CRF++, to learn a morpho-syntactic tagger from this sample. We show how the performance of the learning process can be influenced by different possible decompositions of the target labels into simpler sub-labels.

2 Oral Corpus and its Tagging

This section deals with the morpho-syntactic tagging of an oral corpus, and the difficulties that it causes to the tagger Cordial. The specificities of spoken language lead us to propose a new set of more suitable tags.

The morpho-syntactic Tagging of an Oral Corpus. The purpose of tagging is to assign to each word in a corpus a tag containing morpho-syntactic information about that word. This process can be coupled with stemming, to reduce the occurrence of a given word to its base form or *lemma*. The main difficulty of morpho-syntactic tagging is the ambiguity of words belonging to different lexical categories (*e.g.* the form *portes* in French is either a plural noun (*doors*) or the second person singular of present indicative or subjunctive of the verb *porter* (*to carry*): the tagger must assign the correct tag in a given context. Taggers usually also have problems with words which are not recognized by their dictionary: misspelled words, proper nouns, neologisms, compound words, ...

Tagging an oral corpus faces even more problems. Firstly, the transcriptions are usually not punctuated in order to avoid anticipating the interpretation (Blanche-Benveniste and Jeanjean, 1987 [11]). Punctuation marks such as comma or full stop, and casing are typographical marks. The notion of sentence, mostly graphical, was quickly abandoned by linguists interested in speech. Studies on spoken language have also identified phenomena which are specific to speech, called *disfluence*: repeats, self-corrections, truncations, *etc.* Following (Blanche-Benveniste, 2005 [12]), we believe that all these phenomena should be

[5] `http://crfpp.sourceforge.net/`

included in the analysis of language even if it raises processing issues. Elements, like *hein, bon, bien, quoi, voilà, comment dire*, (*eh, well, what, how to say, ...*) with a high frequency of occurrence in oral corpora, and without punctuation, can be ambiguous, because they can sometimes also be nouns or adjectives (as it is the case for *bon, bien* — meaning *good*). The currently existing tools for tagging are not suitable for oral, which is why this task is so difficult.

The Tagging by Cordial and its Limits. The Socio-Linguistic Survey in Orleans (Enquête Socio-Linguistique à Orléans, ESLO) is a large oral corpus of 300 hours of speech (approximately 4,500,000 words) which represents a collection of 200 interviews recorded in a professional or private context. This investigation was carried out towards the end of the Sixties by British academics in a didactic aim. The corpus is made up of 105 XML files generated by Transcriber, and converted to text format. Each file corresponds to a recording situation. Significant conventions of transcription are:

- the segmentation was made according to an intuitive unit of the type "breathing group" and was performed by a human transcriber;
- the turn-taking was defined by speaker changes;
- no punctuation except exclamation and question marks;
- no upercase letters except named entities;
- word truncation is indicated by the dash (*word-*);
- the transcription is orthographic.

The transcribed data was tagged by Cordial in order to have a reference corpus. This software was chosen for its reliability. As of today, it is one of the best taggers for French written language, with a wide range of tags, rich in linguistic information. The result of tagging is presented in a 3-column format: *word*, *lemma* and *lexical category* (POS), as exemplified in Table 1. Cordial uses about

Word	Lemma	POS
comment (*how*)	comment	ADV
vous (*you*)	vous	PPER2P
faîtes (*make/do*)	faire	VINDP2P
une (*one/a*)	un	DETIFS
omelette (*omelette*)	omelette	NCFS

Table 1. Example of tagging by Cordial.

200 tags encoding morphological information of different kinds such as gender, number or invariability for nouns and adjectives; distinction in modality, tense and person for verbs, and even the presence of aspirated 'h' at the beginning of words. However, the analysis of Cordial's outcome revealed a number of errors. The first group of errors includes "classical" errors such as the following ones.

Ambiguity: *et vous êtes pour ou contre* (*and are you for or against*) ⇒ {contre, contrer, VINDP3S} instead of {contre, contre, PREP3}.

Proper nouns: *les différences qu'il y a entre les lycées les CEG et les CES* (*the differences that exist among [types of secondary and high schools]*) ⇒ {CEG, Ceg, NPMS} instead of {CEG, CEG, NPPIG4} and {CES, ce, DETDEM} instead of {CES, CES, NPPIG}.

Locutions: *en effet* (*indeed*) ⇒ analysed in two separate lines, as opposed to a compound: {en, en, PREP}, then {effet, effet, NCMS} while it is an adverb.

We have also found errors, which are specific to oral data, as :

Truncation: by convention in ESLO, word truncation is indicated by the dash, which raises a problem for tagging by Cordial:
on fait une ou deux réclam- réclamations (*we make one or two complaints*) ⇒ {réclam- réclamations, réclamréclamations, NCMIN5} instead of analysing the sequence in two separate units: {réclam-, reclam-, NCI[6]} and {réclamations, réclamation, NCFP}

Interjection: Cordial does not recognize all the possible interjections present in oral corpora:
alors ben écoutez madame (*so uh listen madam*) ⇒ {ben, ben, NCMIN).
This phenomenon also presents a problem of ambiguity since, according to Dister (2007 [1, p. 350]):

> any form can potentially become an interjection. One, then, observes a grammatical re-classification (...), the phenomenon whereby a word belongs to a grammatical class may, in speech, change.

j'ai quand même des attaches euh ben de la campagne qui est proche quoi (PRI7) (*I still have ties* [euh ben] *to the nearby countryside* [quoi])

Repeat and self-correction: *je crois que le* ({le, le, PPER3S} instead of {le, le, DETDMS}) *le* ({le, le, DETDMS}) *les saisons* (*I think that the the seasons*)

Note, as well, a number of errors such as typos or spellings made by human transcribers. The transcription was not spell-checked.

New Choice of Tags. In order to adapt the tagging to our needs, we propose a number of modifications to the tag set. Those changes are motivated on the one hand by the reduction of the number of tags without loss of the necessary linguistic information, and on the other hand, by the need to adapt the tags to spoken language and to the conventions adopted for the transcription of our corpus. We present here a (non-exhaustive) list of modifications.

- New tags were introduced, such as MI (unknown word) for cases of truncation, and PRES (announcer) for phrases such as *il y a, c'est, voilà* (*there is, this is, there it is*), both very frequent in oral.

[6] Common noun invariable in number

- A few tags, which are too detailed in Cordial, were simplified. For example, the set of tags marking the invariability of adjectives or nouns (masculine invariant in number, feminine invariant in number, singular invariant in gender, plural invariant in number and gender) were replaced by a single tag (*invariable*). The tags marking the possibility for the word to begin with an aspirated 'h' were removed.
- In order to make the system more uniform, some tags were enriched. For example, indications about the gender and number were added to the demonstrative and possessive determiners for coherence purpose with other types, such as definite and indefinite determiners.

The morpho-syntactic tags often contain information of different kinds. They always mark information about the Part-Of-Speech. That basic information seems to be the easiest to acquire from dictionary lookup except, of course, in the case of lexical ambiguity. The tags generally include additional information from different linguistic dimensions:

morphological: concerns the structure of a word as its type and number, the invariability for nouns, adjectives, pronouns and some determiners;
syntactic: describes the function of words in the sentence and they relate with each other, *e.g.* coordination and subordination for conjunctions;
semantic: related to the description of word's meaning such as feature of possessive, demonstrative, definite, indefinite or interrogative for the determiners.

In order to account for that extra information, we propose to structure the tags in 3 levels, called respectively L0 (level of POS tags), L1 (morphological level) and L2 (syntactic and semantic level). A sample of that hierarchical structure is illustrated in Fig. 1. As shown in Fig. 1, some tags:

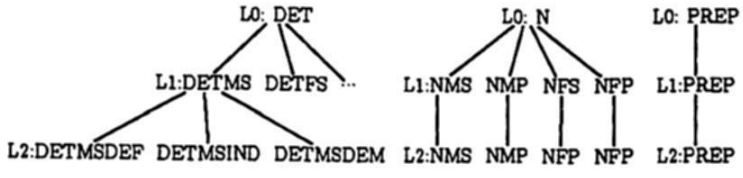

Fig. 1. Sample of the tags hierarchical structure

- remain the same on all 3 levels, *e.g.* adverbs, announcer, prepositions, ...;
- only change on level 2, such as nouns, adjectives, verbs;
- change on every level, including new information such as pronouns and determiners.

In addition to that hierarchical structure, other types of linguistic knowledge can be taken into account during tagging. According to inflectional morphology, a word is made up of a root and a sequence of letters, which often carry morphological information: in French, endings such as *-ait*, *-ais*, *-e* ,*-es* indicate the tense, gender, number, ... In inflectional morphology, those endings are called *grammatical morphemes*. When considering the root as the part shared by all

the forms of a word, it is possible to extract these final sequences from the surface form in order to determine the morphological part of the tag which must be associated with this word. That linguistic knowledge can be exploited in order to improve the performance of a Machine Learning system, as we discuss it in the next section. The reference corpus contains 18,424 words, and 1723 utterances. This data was first tagged by Cordial, and then corrected semi-automatically, in order to make it meet our new tagging conventions. The hand processing was made by linguistic students as part of a 3-month internship.

3 Experiments

We now have a reference corpus, whose labelling was manually corrected and is considered as (nearly) perfect. It is, thus, possible to use it for training a Machine Learning system. As far as learning a tagger is concerned, the best performing statistical model is the one of Conditional Random Fields (CRF) (Lafferty *et al.*, 2001 [3], Sutton and McCallum, 2007 [4]). We choose to work with it. In that section, we first briefly describe the fundamental properties of CRF, then present the experimental process, and finally we detail the results. Our goal is to maximise the use of the linguistic knowledge which guided the definition of our tag set, in order to improve the quality of the learned labels. In particular, we want to determine whether learning the full labels (*i.e.*, those containing all the hierarchical levels of information) could be improved by a sequence of intermediate learning steps involving less information. Note that we do not rely on any dictionary, which would enumerate all the possible labels for a text unit.

CRF and CRF++. CRF are a family of statistical models, which associate an observation x with an annotation y using a set of labelled training pairs (x, y). In our case, each x coincides with a sequence of words, possibly enriched with additional information (*e.g.*, if the words' lemmas are available, x becomes a sequence of pairs (word, lemma)), and y is the sequence of corresponding morpho-syntactic labels. Both x and y are decomposed into *random variables.* The dependencies among the variables Y_i are represented in an undirected graph. The probability $p(y|x)$ of an annotation y, knowing the observation x is:

$$p(y|x) = \frac{1}{Z(x)} \prod_{c \in C} \psi_c(y_c, x) \text{ with } Z(x) = \sum_y \prod_{c \in C} \psi_c(y_c, x)$$

where C is the set of cliques (*i.e.* completely connected subgraph) over the graph, y_c is the configuration taken by the set of random variables Y belonging to the clique c, and $Z(x)$ is a normalization factor. The potential functions $\psi_c(y_c, x)$ take the following form:

$$\psi_c(y_c, x) = \exp\left(\sum_k \lambda_k f_k(y_c, x, c)\right)$$

The functions f_k are called *features*, each one being weighted by a parameter λ_k. The set of features must be provided to the system, whose learning purpose is to

assign the most likely values for each λ_k according to the available valued data. Most of the time, function results are 0 or 1 (but they could also be real-valued).

In linear CRF, which are well-suited to sequence annotation, the graph simply links together the successive variables associated with the sequence elements. The maximal cliques of that kind of graph are, thus, the successive pairs (Y_i, Y_{i+1}). That model is potentially richer than the HMM one, and usually gives better results.

CRF++, the software that we are using, is based on that model. Features can be specified through *templates*, which are instanciated with example pairs (x, y) provided to the program. We kept the default templates provided by the library; they generate boolean functions using the words located within a two-word neighborhood around the current position, as exemplified in Ex. 1.

Example 1. In Table 1, the first column corresponds to the observation x, the third one to the annotation. Hence:
$x =$ "*comment vous faites une omelette*"[7],
$y =$ ADV, PPER2P, VINDP2, PPER2P, DETIFS, NCFS.
For a given position i identifying the clique $(i, i+1)$, the template tests the values of Ys in the clique, and the values of X in position $i, i-2, i-1, i+1, i+2$. At position $i = 3$ we get the following feature f:
if $Y_i =$ VINDP2 and $Y_{i+1} =$ PPER2P and $X_i =$ 'faites' and $X_{i-2} =$ 'comment' and $X_{i-1} =$ 'vous' and $X_{i+1} =$ 'vous' and $X_{i+2} =$ 'une' then $f = 1$ else $f = 0$.

The template also generates simpler functions, where only the positions i, $i-1$, and $i+1$ of X are tested, for example. With that example, we see that the disfluencies are directly taken into account in the model by the fact that they occur in the set of training examples provided to the learner.

Experimental Framework. For our experiments, the corpus was split in 10 subsets and we performed a 10-fold cross-validation. The features are mainly built from observations over words. We have also carried out experiments where the word lemma is supposed to be known. In order to enrich the data even more, we also relied on inflectional morphology, mentioned in Sect. 2:

1. the distinction between *root* and *rest*: the root is the string shared between the word and the lemma, while the rest is the difference between them; if word=lemma, by convention we note Rword=Rlemma=x, else word= Root+Rword and lemma=Root+Rlemma (where the symbol + denotes here the string concatenation);
2. the *word tail*: D_n(word) = n last letters of the word; for instance, if word= 'marchant' and lemma='marcher' then Root='march', Rword='ant', Rlemma ='er' and D_2(word)='nt'.

Reference Experiments. The reference experiments consist of learning the most detailed level (L2) directly. Six different tests were run, which we describe next. We denote by $Feat^{(\text{args})}$ the features built from (args).

Test I $Feat^{(word,\ lemma)}$: *about 10,000,000 features produced;* $F_1 = 0.86$.

Test II $Feat^{(\text{word, lemma, Rword, Rlemma})}$; *11,000,000 features;* $F_1 = 0.88$.

Test III *If* word=lemma *we use* D_2(word) *and* D_3(lemma), *hence:*
$Feat^{(\text{word, lemma, Rword}|_{D_2}(\text{word}),\text{Rlemma}|_{D_3}(\text{lemma}))}$; *20,000,000 feat.;* $F_1 = 0.82$.

Now, if the lemmas are unknown, we obtain the following:

Test IIIbis $Feat^{(\text{word}, D_3(\text{word}))}$; *8,000,000 features;* $F_1 = 0.87$;

Test IV *Similar to III, but with* D_3 *everywhere:*
$Feat^{(\text{word, lemma, Rword}|_{D_3}(\text{word}), \text{Rlemma}|_{D_3}(\text{lemma}))}$; *20,000,000 feat.;* $F_1 = 0.89$.

Again, without relying on lemmas, we get:

Test IVbis $Feat^{(\text{word}, D_3(\text{word}), D_2(\text{word}), D_1(\text{word}))}$; *20,000,000 feat.;* $F_1 = 0.88$.

As expected, the richer the features, the better the results. Knowing the lemmas, for instance, increases the accuracy by 2 points in average. The downside is the increased cost timewise for the learning phase, caused by a much larger number of generated features.

Cascasde Learning. In order to exploit the knowledge contained in the labels — *i.e.*, mainly their organisation in a 3-level hierarchical structure — we first learned each level independently, using the same test set (Test I to IVbis) as previously. The scores obtained are presented in Table 2. We observe that the

Level (num. of tags)	Test I	Test II	Test III	Test IV	Test IIIbis	Test IVbis
L0 (16)	0.93	0.93	0.94	0.94	0.92	0.93
L1 (72)	0.86	0.89	0.9	0.9	0.88	0.89
L2 (107)	0.86	0.88	0.82	0.89	0.87	0.88

Table 2. Accuracy measures when learning separately each of the hierarchical levels of labels.

coarser the levels in terms of how detailed the information is, the easier they are to learn. It can be explained by the reduced number of labels to be learned. Meanwhile, since each level of labels in the hierarchy depends on the previous one, one can hypothesise that using the results from the levels L_j when learning Level L_i (with$j < i$) may improve the results at L_i. The purpose of the next set of experiments is to test that hypothesis: we say that the different hierarchical levels are learned in *cascade*, as in Jousse (2007 [9]) and Zidouni (2009 [10]). In the previous set of experiments, the best scoring tests are Test III and IV; as an attempt to improve those tests, we have designed Test V and VI as follows. We denote by CRF(Li|feat(args)) the learning of level Li knowing the features based on args.

Test V *(derived from Test III)* word, lemma *and* D_3(lemma) *are used to generate the features for learning Level L0; then the result RL0 is used, with the same data, to learn Level L1; and so forth. The successive learning phases are given next.*

- $CRF(L0|feat(\text{word}, \text{lemma}, D_3(\text{lemma}))) \rightarrow ResL0$
- $CRF(L1|feat(\text{word}, \text{lemma}, D_3(\text{lemma}), ResL0)) \rightarrow ResL01$
- $CRF(L1|feat(\text{word}, \text{lemma}, D_3(\text{lemma}), ResL0, ResL01)) \rightarrow ResL012$

Test VI *(derived from TestIV) This time, the initial features are generated with* word, Rword, Rlemma, D_3(word),D_3(lemma). *The successive learning phases are the following:*

- $CRF(L0|feat(\text{word}, \text{Rword}, \text{Rlemma}, D_3(\text{word}), D_3(\text{lemma}))) \rightarrow ResL0$
- $CRF(L0|feat(\text{word}, \text{Rword}, \text{Rlemma}, D_3(\text{word}), D_3(\text{lemma}), ResL0)) \rightarrow ResL01$
- $CRF(L0|feat(\text{word}, \text{Rword}, \text{Rlemma}, D_3(\text{word}), D_3(\text{lemma}), ResL0, ResL01)) \rightarrow ResL012$

Level	III	IV	V	VI
L0	0.94	0.94	0.94	0.94
L1	0.9	0.9	0.88	0.9
L2	0.82	0.89	0.87	0.89

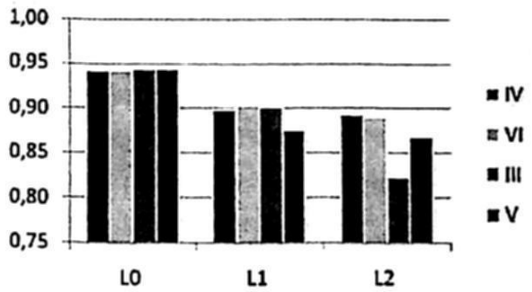

Fig. 2. Accuracy measures for Test III to VI

As shown in Fig. 2, Test V and VI give good results, but not really better than the initial Test III and IV. Therefore, unfortunately, cascade learning does not seem to improve the results obtained in the reference experiments, where L2 is learned directly. That conclusion is confirmed by the experiments without lemmas, the outcome of which we do not detail. Next, we re-consider the way the information contained in the L2 labels is decomposed, in order to better learn those labels.

Learning by Decomposition and Recombination of Labels. We decompose the L2 labels into *components*, so that they can be learned independently. We call *label component* a group of atomic symbols, which cannot all belong to the same label. Intuitively, those components correspond to the possible values for a linguistic feature, such as Gender or Number.

Example 2. The labels in $\mathcal{L} = \{\text{NFS}, \text{NFP}, \text{NMS}, \text{NMP}\}$ come from the concatenation of elements in the sets $\{N\}$ (Noun), $\{M, F\}$ (Gender), and $\{S, P\}$ (Number). All the four labels in $\mathcal{L}$ can be recombined by the cartesian product of three components: $\{N\} \cdot \{M, F\} \cdot \{S, P\}$, where $\cdot$ (dot) denotes the concatenation of sub-labels.

A first option for building those components is to propose the following sets:

- POS={ADJ, ADV, CH, CONJCOO, CONJSUB, DET, INT, INT, MI, N, PREP, PRES, P, PP, V}
- Genre={M, F}; Pers={1, 2, 3}; Num={S, P}
- Mode_Tense={CON, IMP, SUB, IND, INDF, INDI, INDP, INF, PARP, PARPRES}
- Det_Pro={IND, DEM, DEF, POSS, PER, INT}

Each of those components can be learned independently. However, some of them are still mutually exclusive: for example, Person and Gender can be grouped together since their values (respectively in $\{1, 2, 3\}$ and $\{M, F\}$) never occur together. On the contrary, Gender and Number can not be grouped, since, for instance, the value 'F' may occur with 'S' or 'P' within the same label. We end up working with 4 components: $G0$ = POS, $G1$ = Genre $\cup$ Pers $\cup \{\epsilon\}$, $G2$ = Num $\cup \{\epsilon\}$, and $G3$ = Mode_Tense $\cup$ Det_Pro $\cup \{\epsilon\}$. with ϵ the empty string, neutral element for the concatenation. Each of these label subsets can now be learned independently by a specific CRF. In that case, the final label proposed by the system results from the concatenation of all the CRF outcomes. If we denote by $\cdot$ (dot) the concatenation operator, the cartesian product $G0 \cdot G1 \cdot G2 \cdot G3$ generates every L2 label. But it also generates labels which are not linguistically motivated. For example, ADVMP = ADV $\cdot$ M $\cdot$ P $\cdot \epsilon$ is meaningless, since an adverb is invariable. In order to avoid that problem, we have tested two different approaches. The first one consists of using a new CRF, whose features are the components learned independently. The second one consists of intoducing explicit symbolic rules in the concatenation process, in order to rule out the illegal combinations. Example rules are as follows:

- ADV, CONJCOO, CONJSUB and INT can only combine with ϵ
- V can not combine with values of Det_Pro
- DET can not combine with values of Mode_Tense

Those rules exploit the fact that the POS category (*i.e.* the G0 component) is learned with strong enough confidence ($F_1 = 0.94$) to constrain the other sub-labels with which it may combine. We have carried out the tests presented below:

Expe. 1 $CRF(G_i|feat(\text{word}, \text{lemma}, D_3(\text{word}))) \rightarrow ResGi$

We have also tested different versions where feature generation is achieved without lemma but with word tails instead (as in Test IVbis).

Expe. 2 $CRF(G_i|feat(\text{word}, D_3(\text{word}), D_2(\text{word}), D_1(\text{word}))) \rightarrow ResbisGi$

Test VII $CRF(L2|feat(\text{word}, \text{Rlemma}, ResG0, ResG1, ResG2, ResG3)) \rightarrow ResL2$

Test VIIbis *Same as VII, but without lemmas:*
$CRF(L2|feat(\text{word}, ResbisG0, ResbisG1, ResbisG2, ResbisG3)) \rightarrow ResbisL2$

Test VIII *Here, the outcome from Test VII is replaced by symbolic combination rules using the results ResGi.*

Test VIIIbis *Same as VIII, except that the combination rules use ResbisGi.*

Figure 3 illustrates the two possible strategies for recombining the full labels, along with the results from learning each component independently (accuracy measures). Note that the component G2 is better learned without lemmas but

Component	ResGi	ResbisGi
G0	0.94	0.92
G1	0.92	0.92
G2	0.93	0.95
G3	0.95	0.94

Test	VII	VIIbis	VIII	VIIIbis
F_1	0.89	0.875	0.9	0.895

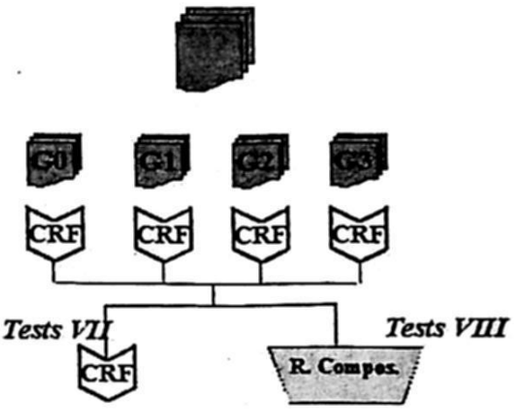

Fig. 3. Learning components: accuracy measures for two recombination strategies.

with word tails. The recombination strategy based on CRF (Test VII and VIIbis) does not improve the scores obtained by direct learning of the full labels on L2 (Test IV and IVbis). However, the rule-based recombination strategy (Test VIII and VIIIbis) does improve direct learning. Test VIIIbis illustrates that, in general, the absence of lemmas can be balanced by word tails associated with symbolic recombination of the labels. Meanwhile, timewise the learning phase is considerably improved by the recombination strategy: Test VIII only takes 75 min., while Test IV takes up to 15h. (using a standard PC). It should also be noted that since the labels obtained by recombination are most of the time only partially (in)correct, those experiments would be better evaluated with other measurements than accuracy.

Note, as well, that the definition we provide of a specific set of labels prevents comparing our performances against those of other taggers.

4 Conclusion

In that paper, we have shown that it is possible to efficiently learn a morpho-syntactic tagger specialised for a specific type of corpus. First, we have seen that the specificities of spoken language are difficult to enumerate. Instead of trying to rule them all, it is natural to rely on Machine Learning techniques. Our experiments all take the input data as they are, without filtering out any difficulties.

Note that it is not possible to rigourously compare the performance achieved by Cordial with the performances reported here, since the target label sets are different. Yet, the performances of the best learned taggers seem comparable to those usually obtained by Cordial on oral corpora. The insentive of using CRF for that task is that it does not require many parameters to be set, and that the settings involved are flexible enough to integrate external linguistic knowledge. We have mostly used here our understanding of the labels in order to focus on learning sub-labels, which are simpler and more coherent. The performance would have also certainly been improved by the use of a dictionary of labels for each word, or each lemma, to specify features.

Finally, it seems quite difficult to still improve the quality of the learned labels by simply relying on the decomposition in simpler sub-labels. However, that strategy by decomposition is very efficient timewise, and the learning process has been greatly improved in that respect. It is also interesting to notice that the most efficient strategy relies on a combination between statistic learning and symbolic rules. Further works are going in that direction.

References

1. Dister, A.: De la transcription à l'étiquetage morphosyntaxique. Le cas de la banque de données textuelle orale VALIBEL. Thèse de doctorat, Université de Louvain (2007)
2. Valli, A., Veronis, J.: Étiquetage grammatical des corpus de parole : problèmes et perspectives. Revue française de linguistique appliquée **IV**(2) (1999) 113–133
3. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of ICML'01. (2001) 282–289
4. Sutton, C., McCallum, A. In: 1 An Introduction to Conditional Random Fields for Relational Learning. The MIT Press (2007)
5. McCallum, A., Li, W.: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In: Proceedings of CoNLL. (2003)
6. Pinto, D., McCallum, A., Lee, X., Croft, W.: Table extraction using conditional random fields. In: SIGIR'03: Proceedings of the 26th ACM SIGIR. (2003)
7. Altun, Y., Johnson, M., Hofmann, T.: Investigating loss functions and optimization methods for discriminative learning of label sequences. In: Proceedings of EMNLP. (2003)
8. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: Proceedings of HLT-NAACL. (2003) 213–220
9. Jousse, F.: Transformations d'abres XML avec des modèles probabilistes pour l'annotation. Thèse de doctorat, Université de Lille (2007)
10. Zidouni, A., Glotin, H., Quafafou, M.: Recherche d'Entités Nommées dans les Journaux Radiophoniques par Contextes Hiérarchique et Syntaxique. In: Actes de Coria. (2009)
11. Blanche-Benveniste, C., Jeanjean, C.: Le Français parlé. Transcription et édition, Paris (1987)
12. Blanche-Benveniste, C.: Les aspects dynamiques de la composition sémantique de l'oral. In Condamines, A., ed.: Sémantique et corpus. Lavoisier, Hermès, Paris (2005) 39–74

Chinese Named Entity Recognition with the Improved Smoothed Conditional Random Fields[1]

Xiaojia Pu, Qi Mao, Gangshan Wu[2], and Chunfeng Yuan

Department of Computer Science and Technology, Nanjing University
{puxiaojia, maoq1984}@gmail.com, {gswu, cfyuan}@nju.edu.cn

Abstract. As a kind of state-of-the-art sequence classifier, Conditional Random Fields (CRFs) recently have been widely used for some natural language processing tasks which could be viewed as the sequence labeling problems such as POS tagging, named entity recognition(NER) etc. But CRFs suffer from the failing that they are prone to overfitting when the number of features grows. For NER task, the feature set is very large, especially for Chinese language, because of it's complex characteristics. Existing approaches to avoid overfitting include the regularization and feature selection. The main shortcoming of these approaches is that they ignore the so-called unsupported features which are the features appearing in the test set but with zero count in the training set. Actually, without the information of them, the generalization of the CRFs suffers. This paper describes a model called Improved Smoothed CRF which could capture the information of the unsupported features using the smoothing features. It provides a very effective and practical way to improve the generalization performance of CRFs. Experiments on Chinese NER proved the effectiveness of our method.

Keywords: Chinese named entity recognition; Conditional Random Fields; overfitting; generalization; Improved Smoothed CRF; smoothing feature

1 Introduction

Named entity recognition (NER) is one of the fundamental works in natural language processing and text processing. The aim of NER is to find the names of some special entities from the free texts, e.g. person, location, organization etc.

Compared with English, Chinese NER is more difficult because of its complex characteristics. For example, a sentence of English is a sequence of words, and the words will be separated by the space, but in Chinese, the sentence is a sequence of characters without any spaces between them.

Viewed as a sequence labeling problem, various sequence labeling models have been used to solve the NER problem such as Hidden Markov model (HMM) [1], Maximum Entropy (ME) [2], Maximum Entropy Markov model (MEMM) [3],

[1] This work was supported by the research grants from the Natural Science Foundation of China (60721002 and 60975043).

[2] Corresponding author.

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 91-103

Received 05/12/09
Accepted 16/01/10
Final version 12/03/10

Conditional Random Fields (CRFs) [4-10],and so on. Many works have proven that CRFs get the excellent performance and are superior to the previous models [13].

A key advantage of CRFs is that they can support the use of complex features, e.g. long-range features that are overlapping and inter-dependent. This flexibility encourages the use of a rich collection of features. But with a large feature set, the CRFs are prone to overfitting [13] [15], e.g. in the Chinese NER task, because of the complex characteristics, the scale of the features will be more than millions. So it's really important to solve the overfitting problem.

There are two existing approaches to address this problem: regularization [12] [14] and feature selection [11] [15]. The regularization method is adding a regularization term to the objective function to penalize the large weight vectors. This method is also called smoothing [13] [20], similar with the smoothing methods in Maximum Entropy model [19].A typical feature selection approach is the feature induction [11], which induces the best features from the candidate sets in an iterative and approximate manner.

We conducted a detailed analysis of the issues which could influence the generalization ability of CRFs. After the study of these existing approaches, we found that the main disadvantage of these approaches is that they didn't take into account the so-called unsupported features [12], which were with zero count in the training set but occurred in the test set. Surely, these unsupported features will have an important impact on the generalization of CRFs [12]. For example, in the named entity recognition task, due to the lack of some efficient features which just appear in the test data set, some named entities could always be very difficult to recognize.

In this paper, we propose a new model called Improved Smoothed CRF, adopting a new smoothing method to help improve the generalization ability of CRFs. The insight of our method is that though the unsupported features are unknown, but we could use some high-level features to cover them. These high-level features, called smoothing features, are predefined based on the feature templates. Each smoothing feature is corresponding to a feature template. During the training procedure, in order to estimate the distribution of these smoothing features, a validation set, which is divided from the whole training set randomly, will be used to simulate the test set. Then, the ordinary features and smoothing features will be integrated together for training. During the decoding procedure, the unknown feature will all be mapped into the corresponding smoothing feature which will keep the information of unknown feature, rather than just omitting it.

In order to evaluate our method, we did some comparative experiments on Chinese NER task, and the result showed its effectiveness. Besides, we try to analyze why this method is efficient, and have a discussion about this.

The contribution of our work mainly includes: (1) a detailed analysis of the issues which influence the generalization of CRFs and some existing approaches to improve the generalization ability; (2) propose an Improved Smoothed CRF and show its effectiveness on Chinese NER task.

The paper is organized as follows. In Section 2, we will review Conditional Random Fields and analyze its generalization ability. In Section 3, we will have a detailed description about our Improved Smoothed CRF. Consequently, the experiment and the result analysis will be arranged in Section 4. Finally, we will give the conclusion and some possible future works.

2 Conditional Random Fields and Analysis of its Generalization

Conditional Random Fields (CRFs) are a class of discriminative probabilistic models trained to maximize a conditional probability. A common used special graph structure is a linear chain as shown in fig.1 and it avoids the label biased problem of Maximum Entropy Markov Models (MEMM) [3].

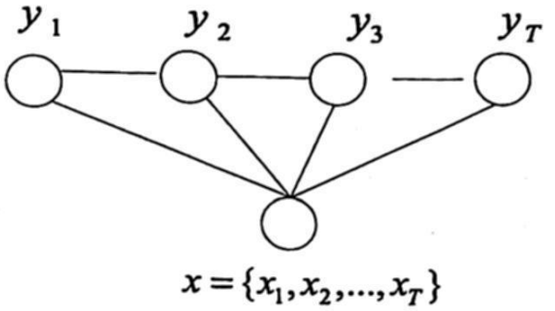

Fig. 1. A linear-chain CRF model

A linear-chain CRF[3] with parameters $\Lambda=\{\lambda_1,\lambda_2,...,\lambda_T\}$ defines a conditional probability for a state sequence $y=\{y_1,y_2,...,y_T\}$ given the observation sequence $x=\{x_1,x_2,...,x_T\}$ be the

$$p_\Lambda(y\mid x)=\frac{\exp(\sum_t\sum_k \lambda_k f_k(y_{t-1},y_t,x,t))}{Z(x)}, \tag{1}$$

where $Z(x)$ is the normalization constant that makes the probability sum to one. f_k is a feature function which is often binary-valued, and λ_k is a learned weight associated with feature f_k. Feature functions can measure any aspect of the a state transition, $y_{t-1}->y_t$,and the observation sequence, x ,centered at the current time step, t .For example, one feature function might have the value 1 when y_{t-1} is the sate B, y_t is the state I, and x_i is the character '国'.

2.1 Training of CRF

The training will be finished by maximizing the log-likelihood L_Λ on the given training set $D=\{(x_1,y_1),(x_2,y_2),...,(x_N,y_N)\}$.

[3] For convince to describe our work, the CRF mentioned in this paper will all be linear-chain CRF.

$$\tilde{\Lambda} = \arg\max_{\Lambda \in R^k} L_\Lambda \tag{2}$$

where

$$L_\Lambda = \sum_{i=1}^{N}\left(\sum_t \sum_k \lambda_k f_k(y_{t-1}^i, y_t^i, x^i, t) - \log Z(x^i)\right). \tag{3}$$

Because the likelihood function is convex, we can get the optimization by seeking the zero of the gradient, i.e. the partial derivation

$$\frac{\partial L_\Lambda}{\partial \lambda_k} = \sum_{i=1}^{N}\sum_t f_k(y_{t-1}^i, y_t^i, x^i, t) - \sum_{i=1}^{N}\sum_t \sum_y \left(p_\Lambda(y \mid x^i) f_k(y_{t-1}, y_t, x^i, t)\right). \tag{4}$$

The first term is the expected value of f_k under the empirical distribution $\tilde{p}(x,y)$. The second term is the expectation of f_k under the model distribution $p(y \mid x)$. For the easy understanding, the formula (4) could be written as

$$\frac{\partial L_\Lambda}{\partial \lambda_k} = E_{\tilde{p}(x,y)}[f_k] - E_{p_\Lambda(y|x)}[f_k], \forall k . \tag{5}$$

Therefore, for the maximum likelihood solution, when the gradient is zero, the two expectations are equal. And setting this gradient to zero does not result in any closed form solution, so it typically resorts to iterative methods, such as the L-BFGS [16], which has been the method of choice since the work of [17].

2.2 Analysis of the Generalization

In this section, we try to conduct a detailed analysis of the issues which will lead to the overfitting problem and decrease the generalization ability of CRF. These issues include:

Firstly, the method of CRF training can be viewed as maximum likelihood estimation (MLE), and like other maximum likelihood methods, this type of modeling is prone to overfitting, because of its inherent weaknesses, i.e. without any prior information about the parameter distribution [18]. The common method to avoid this is using the Maximum a Posterior (MAP) method instead.

Secondly, formula (1) tells us that CRF is an exponential linear model. The scale of the parameter is equal to the number of features. With the increasing of features, the parameter dimension will be very large, and thus, the freedom of parameters will be enlarged. For Chinese NER, the scale of the features reached millions or more than millions, and almost 35% of them are sparse features. In order to fit these sparse features, some parameters will become very large. This highly uneven distribution of the parameter values will lead to the overfitting problem.

Further, during the original CRF training, usually, the features used are simply aggregated from the training set following the feature templates. But, the diversity between the training set and test set is inevitable, the features occurred in the training

set are not able to cover the full feature set. So, during the decoding period, the original CRF will omit the unsupported features [12], the features occurred in the test set but with zero count in the training set, ignoring the fact that these features surely contain the useful information. In [12], they found that the unsupported features can be extremely useful for pushing Viterbi inference away from certain paths by assigning such features negative weight. So if a model trained without the information of the unsupported features, when applied to the test set, of course, the generalization ability will decrease.

2.3 Existing Work

Focusing on part of the issues we talked above, there have been some approaches to avoid the overfitting problem, including feature selection and regularization.

Feature Selection. Wise choice of features is always vital in machine learning solutions. For CRF, this method aims at the second issue we talked above to reduce the parameter dimension. Typical method for the feature selection is feature induction [11] which induces the best features from the candidate sets in an iterative and approximate manner. But the computing cost will be very large when the scale of the features grows, so it's not very practical in some applications such as Chinese NER.

Regularization. As a common way to avoid overfitting, regularization is a penalty on the parameter vectors whose norm is too large. Instead of maximizing solely the likelihood of the training set, typically, a quadratic penalty term is added

$$L_\Lambda = \sum_{i=1}^{N}\left(\sum_t\sum_k \lambda_k f_k(y_{t-1}^i, y_t^i, x^i, t) - \log Z(x^i)\right) - \sum_k \frac{\lambda_k^2}{2\delta^2}, \tag{6}$$

where δ^2 specifies how much the penalty is applied.

In general, the penalty prevents the absolute values of parameters $\{|\lambda_k|\}$ from becoming too large. And this method has a Maximum a Posterior (MAP) interpretation that the parameter Λ follows a Gaussian prior distribution.

Following the expression of formula (5), the gradient of the objective function could be written as:

$$\frac{\partial L_\Lambda}{\partial \lambda_k} = E_{\tilde{p}(x,y)}[f_k] - E_{p_\Lambda(y|x)}[f_k] - \frac{\lambda_k}{\delta^2}, \forall k\,. \tag{7}$$

The regularization method is also called smoothing [13] [15] [20], because it is similar with the smoothing methods in the Maximum Entropy model [19]. It aims at the first and second issues we talked above, by adapting a MAP training method and tighten the values of the parameters.

So we can conclude that the existing approaches all ignored the third issues, i.e. the unsupported features, actually, the unsupported features have a great impact on the generalization performance of the CRF.

An intuitive approach is that we numerate the full feature set, containing all the possible features, and then use the regularization (smoothing) method. Then during the training, all the unsupported features will get the non-zero weight. However, for Chinese NER, it's hard to numerate the full feature set due to the high scale of Chinese characters and the complex characteristics of language, and besides, doing so often greatly increases the number of parameters which will cause the overfitting.

[12] presents a compromising method called incremental support, which will introduce just some heuristic unsupported features in a iterative way. However, it's not practical for the large feature set of Chinese NER.

3 Improved Smoothed Conditional Random Fields

We propose a new model called Improved Smoothed Conditional Random Field, which provides a practical way to capture the information of unsupported features, by the means of inducing the smoothing features.

The inspiration of our method is that though the unsupported features are unknown, but we could use some high-level features to cover them. Every high-level feature, called smoothing feature, is predefined corresponding to a feature template. Since the unsupported feature is also generated based on a special feature template, we could use the smoothing feature to replace it.

During the training, in order to extract the smoothing features with the estimation of the distribution of them, we use a validation set as the simulation of test set. The validation set is divided from the whole training set randomly. And in the end, the ordinary features and smoothing features will be put together for the training. During the decoding, the unknown feature will all be mapped to the corresponding smoothing feature rather than solely omitted.

The advantage of our method is that we provide a practical way to capture the information of unsupported features, and meanwhile, the parameter dimension will not be enlarged too much because the smoothing feature set is small.

3.1 Smoothing Features

The so-called smoothing features are predefined based on the feature templates, as shown in Table 2.

For a given training set *Train_set*, after the feature selection, we could get a feature vector $F_1 = \{f_1, f_2, ..., f_K\}$, and for a given validation set *V_set*, following the same feature selection method, we could get a feature vector $F_2 = \{f_1', f_2', ..., f_m'\}$.we use the $T(f)$ to represent the template which generates the feature f. If $f \in F_2$ and $f \notin F_1$, we use a special feature $f^*(T)$ to describe $T(f)$, and $f^*(T)$ is called the smoothing feature for a given feature template.

3.2 Extraction of Features

For the training of CRF, the extraction of features is vital as the first step, including calculating the frequencies of each feature. The extraction sequence for the Improve Smoothed CRF is shown in Fig.2.

Different with the original CRF, during the extraction procedure, the Improved Smoothed CRF will divide the features into ordinary features and smoothing features. As shown in Fig. 2, the training set D is divided into two subsets, D_1 and D_2. D_1 is the data set, and D_2 is the validation set. F_3 is the collection of the smoothing features, and $F*$, containing the distribution of smoothing features, is the final smoothing feature set. Actually, the validation set is used to simulate the test set to get distribution of smoothing features. F_1 and $F_2 - F*$ are the ordinary features. F is the final full feature set with ordinary features and smoothing features including the distribution of them.

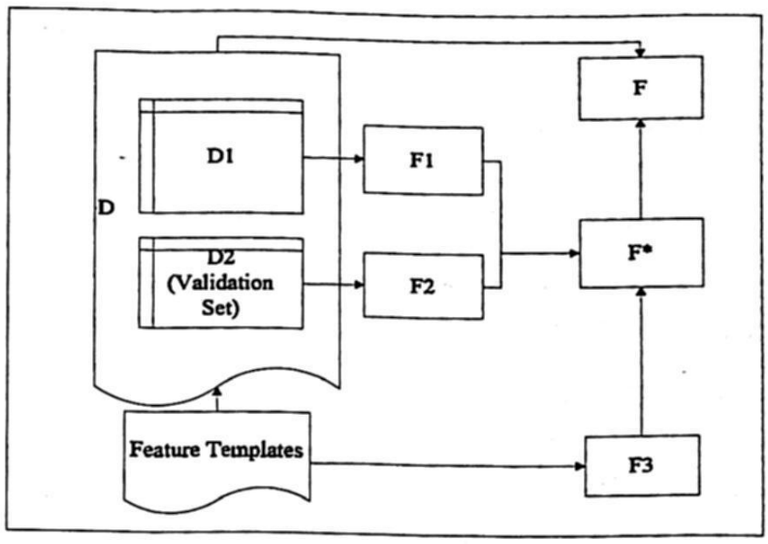

Fig. 2. The sequence of extracting features

3.3 The Definition of Improved Smoothed CRF

Following the definition of original CRF, the conditional probability of the state sequence $y=\{y_1, y_2, ..., y_T\}$ given the observation sequence $x=\{x_1, x_2, ..., x_T\}$ defined by the Improved Smoothed CRF could be formulized as

$$p(y|x)=\frac{\exp\left(\sum_t\left[\sum_k \lambda_k f_k(y_{t-1}, y_t, x, t)+\sum_m \mu_m g_m(y_{t-1}, y_t, x, t)\right]\right)}{Z(x)} \tag{8}$$

where $g_m(y_{t-1}, y_t x, t) \in F*$ is the smoothing feature at the time t, μ_m is the weight associated with the feature.

From formula (8), it seems that the new model is the same as the original CRF, but essentially, they are different. The new model use the predefined smoothing features

to cover the unsupported features, and predict the distribution of unsupported features based on the validation set which is the simulation of the test data.

Incorporating the smoothing features, the model will more fit to the test data, and improve the generalization performance.

Because it can be seen as an extension of regularization (smoothing) in an improved smoothing way, we call it the Improved Smoothed CRF.

3.4 Training of the Improved Smoothed CRF

After the extraction of the features, the training procedure is the same as the original CRF, the entire training process is shown blow:

Algorithm: The training of Improved Smoothed CRF

```
Improved Smoothed CRF training (D,F_temp)
  Input: D={d_i},di={x_i,y_i},F_temp  //D is the training set, F_temp
are the feature templates
  Output: Λ={λ_1,...,λ_k,μ_1,...,μ_m}  //Λ is the weights of the
parameters, λ_1,...,λ_k for the ordinary features, μ_1,...,μ_m for the
smoothing features
begin
    divide D into D_1 and D_2 (validation set)
    extracting F_1, F_2 from D_1, D_2
    generate F_3 based on F_temp
    extracting F* with F_1, F_2 and F_3
    Λ= 0;//the initialization
    do {
    for d={x,y} in D do
       calculate the p_Λ(y|x) of d based on formula(1),
       for λ/μ in the features of d do
           update ∂L_Λ/∂λ(μ) based on formula(6)
       end do
       update L_Λ based on p_Λ(y|x)
    end do
    L-BFGS(Λ,L_Λ,{∂L/∂λ(μ)_i})
    //update the parameters of L-BFGS
    }until(converged)
end.
```

3.5 Decoding of the Improved Smoothed CRF

The decoding process is the same as the original CRF after replacing the unsupported features with the smoothing features.

4 Experiments

We did comparative experiments on Chinese NER task to evaluate our method in two different corpuses, MSR [21] and PKU [22]. We pick up 3000 sentences separately from the two corpuses as the test sets, the remains are used for training.

Chinese NER problem can be solved mainly in two levels: word level [8] [9] and character level [6] [10].In order to analyze the results clearly, we did our experiment in the character level.

We extend the CRF4 in Mallet toolkit [23] to implement our improved smoothed CRF, and the baseline model is CRF4, which is an implementation of original CRF.

The evaluation metrics is precision, recall and F measure.

4.1 Feature Templates and Tag Set

The feature templates are shown in Table 1. Since the purpose of our experiments is to compare the performance between the original CRF and Improved Smoothed CRF, so the features templates we used are the basic and simple ones, which will make the analysis of the results more clearly and persuasive.

Table 1. Feature templates used in the experiment

type	template
Base feature	$C_n(n=-2,-1,0,1,2)$
Bi-gram feature	$C_nC_{n+1}(n=-2,-1,0,1)$
	$C_{-1}C_1$

C_n is the character with the relative distance n from the observation position, these feature templates are the basic templates for Chinese NER.

Corresponding to the feature templates, some smoothing features are listed in Table 2. We just list the node features without the combination with the state transition features, actually during the training, the state transition should be considered.

Table 2. Some corresponding smoothing features

template	Smoothing features
C2	<Unknown>@2
C-1C0	<Unknown>_-1&_<Unknown>@0
C1C2	<Unknown>@1_&_<Unknown>@2
…	…

The tag set is the coding of the states, and we choose BIO as our tag set, the full states are $\{O, B-NR, I-NR, B-NS, I-NS, B-NT, I-NT\}$.

NR, NS and NT represent the name, location and organization respectively. O represents that it's not the named entity, and $B-NR$ represents that it's the first

character of the named entity, $I-NR$ represents that it's the character of the named entity but not the first. The NS and NT are similar.

4.2 Comparative Results

The comparative experiments results are shown in Table 3 and Table 4.The three types of named entities to be recognized are person, location and organization.

Table 3. Comparative results based on PKU corpus

	Metric	Person	Loc.	Org.	All
Original CRF	Precision	92.83%	89.62%	85.21%	90.26%
	Recall	80.69%	81.45%	77.54%	80.13%
	F-measure	86.33%	85.34%	81.19%	84.89%
Imp. Smoothed CRF	Precision	93.05%	90.72%	87.66%	91.28%
	Recall	87.96%	85.38%	81.46%	85.90%
	F-measure	90.43%	87.97%	84.44%	88.51%

Table 4. Comparative results based on MSR corpus

	Metric	Person	Loc.	Org.	All
Original CRF	Precision	93.95%	86.02%	82.08%	87.44%
	Recall	72.14%	74.14%	64.25%	70.79%
	F-measure	81.62%	79.64%	72.08%	78.24%
Imp. Smoothed CRF	Precision	92.69%	87.72%	81.99%	87.84%
	Recall	82.36%	79.02%	70.49%	77.78%
	F-measure	87.22%	83.15%	75.81%	82.50%

For the experiments in PKU corpus, we used about 16 thousand sentences for training, and in MSR corpus, we used about 20 thousand sentences.

Because we aim to compare the performance of the models rather than get the best recognition result, so we didn't use the entire corpus for training. The scale of our validation set is about the 1/3 to 1/2 of the training corpus.

We can find that with the Improved Smoothed CRF, the F measure improved in both corpuses, e.g. 3.62% in PKU, 4.26% in MSR.

The recall got the largest increase, e.g. 5.77% in PKU, 6.99% in MSR. This increase indicates that the information of unsupported features is very useful, and our model could capture them efficiently. With this information, we could recognize some entities which couldn't be recognized correctly using the original model.

4.3 The Change of the Parameters

In order to know clearly about the impact on the parameters by the Improved Smoothed CRF, some values of the parameters $f(y_{t-1}, y_t, \text{'国'})$ are shown in Table 5.

Table 5. The Change of some example parameters

Features(国)	Original CRF	Imp. Smoothed CRF
O -> O	0.27270093137463947	-0.09193322956278925
O -> B-ns	0.5236346196173783	0.3431855304817065
B-nr -> O	-0.18969131328341626	0.11111461613932119
B-nr -> I-nr	0.14558200966493068	0.12232168710742694
B-nr -> B-ns	0.05574692403247397	-0.11382501213178316
B-nr -> B-nt	0.03156266967717719	0.11615382524919472
...	...	...

We can see that the parameters become more tightly, which could help to improve the generalization performance and the experiment result in deed showed this.

4.4 Different Training Size

We also did an experiment in MSR corpus to find how the size of training set influences the performance. The result is shown in fig. 3.

The 4 training set we used are respectively 25%, 50%, 75%, 100% of the whole corpus, for the 75% and 100% of the corpus, due to the limit of the memory of JVM and our machine, we discarded the sparse features, but the experiment result still proved that our model could improve the performance.

With the increase of the training scale, the improvement reduced relatively. This indicated that the unseen information of the unsupported features is more useful for the small training set, and our model could deal with this effectively.

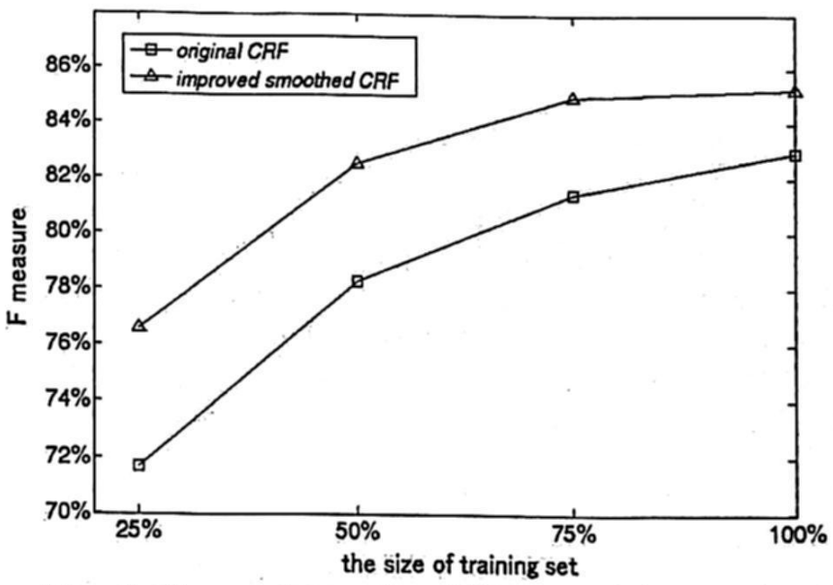

Fig. 3. The result based on different training set size

4.5 Analysis and discussion

The most particulars of our smoothing method are that they could capture some unknown features, and also could reduce the computation cost. We think that, for NER task, the sparse unknown features have something in common, and our method could make the best use of the common characteristics.

Anyway, for Chinese NER, this special task, our smoothing method give a very effective and practical way to improve the generalization of CRF. And in the future works, we want to do some experiments in other tasks to analyze that whether this smoothing method is limited to some special tasks or depends on factors like text genre or text domain.

5 Conclusion

In this paper, we take a detailed analysis of the factors influencing the generalization ability, and then we propose an Improved Smoothed CRF. The substance of our work is that using smoothing features to capture the information of unsupported features and using the validation set to simulate the test set.

This Improved Smoothed CRF provides a practical and effective way to increase the generalization performance of CRF. The experiments on Chinese NER proved its effectiveness. We will incorporate more meaningful feature templates such as surname list, location ending list, etc. in [5-10] to achieve a better result for the Chinese NER.

And we believe that our method could also be useful in other NLP tasks, e.g. POS tagging, Chinese word segmentation, etc.

References

1. L. R. Rabiner: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In: Proceedings of IEEE, pp. 257--285. (1989)
2. A. L. Berger, S. A. D. Pietra, V. J. D. Pietra: A Maximum Entropy Approach to Natural Language Processing. Computational Linguistics, pp. 39-71. (1996)
3. A. McCallum, D. Freitag, F. Pereira: Maximum Entropy Markov Models for Information Extraction and Segmentation. In: Proceedings of ICML'2000.
4. J. Lafferty, A. McCallum, F. Pereira: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proceedings of ICML' 2001, pages 282-289.
5. A. McCallum, W. Li: Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons. In: Proceedings of 7th Conference on Natural Language Learning (CoNLL). (2003)
6. W. Chen, Y. Zhang, H. Isahara: Chinese Named Entity Recognition with Conditional Random Fields. In: Proceedings of the 5th SIGHAN Workshop on Chinese Language Processing & the 3rd International Chinese Language Processing Bakeoff. (2006)
7. A. Chen, F. Peng, R. Shan, G. Sun: Chinese Named Entity Recognition with Conditional Probabilistic Models. In: Proceedings of the 5th SIGHAN Workshop on Chinese Language Processing& the 3rd International Chinese Language Processing Bakeoff. (2006)

8. Z. Xu, X. Qian, Y. Zhang, Y. Zhou: CRF-based Hybrid Model for Word Segmentation, NER and even POS Tagging. In: The 4th International Chinese Language Processing Bakeoff & the First CIPS Chinese Language Processing Evaluation. (2008)
9. H. Zhao, C. Kit: Unsupervised Segmentation Helps Supervised Learning of Character Tagging for Word Segmentation and Named Entity Recognition. In: The 4th International Chinese Language Processing Bakeoff & the First CIPS Chinese Language Processing Evaluation. (2008)
10. Yuanyong Fen, Le Sun, Wenbo Li, Dakun Zhang: A Rapid Algorithm to Chinese Named Entity Recognition Based on Single Character Hints. Journal of Chinese Information Processing, Vol.22, No.1, pp.104-110. (2008)
11. A. McCallum: Efficiently Inducing Features of Conditional Random Fields. In: Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI). (2003)
12. F. Peng, A. McCallum: Accurate Information Extraction from Research Papers using Conditional Random Fields. In: Proceedings of Human Language Technologies: The 11thAnnual Conference of the North American Chapter of the Association for Computational Linguistics. (2004)
13. Roamn Klinger, Katrin Tomanek: Classical Probabilistic Models and Conditional Random Fields. Algorithm Engineering Report TR07-2-013, Dortmund University of Technology. (2007)
14. Charles Sutton, Andrew McCallum: An introduction to Conditional Random Fields for Relational Learning. In: Lise Getoor, Benjamin Taskar (Editors.): Introduction to Statistical Relational Learning, MIT Press, Chap.1, pp.93-127. (2006)
15. Trevor A. Cohn: Scaling Conditional Random Fields for Natural Language Processing. Ph.D Thesis, University of Melbourne. (2007)
16. D.C. Liu, J. Nocedal: On the Limited Memory BFGS Method for Large Scale Optimization. Mathematical Programming, pp. 49-55. (1989)
17. F. Sha, F. Pereira: Shallow Parsing with Conditional Random Fields. In: Proceedings of Human Language Technologies: The 11thAnnual Conference of the North American Chapter of the Association for Computational Linguistics. (2003)
18. Stanley F. Chen, Rosenfeld Ronald: A Survey of Smoothing Techniques for ME Models. In: IEEE Transactions on Speech and Audio Processing, Vol.8, No.1, pp. 37-50. (2000)
19. Stanley F. Chen, Rosenfeld Ronald: A Gaussian Prior for Smoothing Maximum Entropy Models. Technical Report CMU-CS-99-108, Carnegie Mellon University. (1999)
20. D. L. Vail, J. D. Lafferty, M. M. Veloso: Feature Selection in Conditional Random Fields for Activity Recognition. In: Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, Oct 29- Nov 2.(2007)
21. Corpus of MSR, http:// www.sighan.org/bakeoff2006/
22. Corpus of Peking University, http://icl.pku.edu.cn/icl_groups/corpus/dwldform1.asp
23. Mallet Toolkit, http://mallet.cs.umass.edu

Ontology-Driven Approach to Obtain Semantically Valid Chunks for Natural Language Enabled Business Applications

Shailly Goyal, Shefali Bhat, Shailja Gulati, C Anantaram

Innovation Labs, Tata Consultancy Services Ltd
Gurgaon, India

Abstract. For a robust natural language question answering system to business applications, query interpretation is a crucial and complicated task. The complexity arises due to the inherent ambiguity in natural language which may result in multiple interpretations of the user's query. General purpose natural language (NL) parsers are also insufficient for this task because while they give syntactically correct parse, they lose on the semantics of the sentence. This is because such parsers lack domain knowledge. In the present work we address this shortcoming and describe an approach to enrich a general purpose NL parser with domain knowledge to obtain semantically valid chunks for an input query.
A part of the domain knowledge, expressed as domain ontology, along with the part-of-speech (POS) tagging is used to identify the correct predicate-object pairs. These pairs form the constraints in the query. In order to identify the semantically valid chunks of a query, we use the syntactic chunks obtained from a parser, constraints obtained by predicate-object binding, and the domain ontology. These semantically valid chunks help in understanding the intent of the input query, and assist in its answer extraction. Our approach seamlessly works across various domains, given the corresponding domain ontology is available.

1 Introduction

Natural language (NL) enabled question answering systems to business applications [1, 2] aim at providing appropriate answers to the user queries. In such systems, query interpretation is a fundamental task. However, due to the innately ambiguous nature of the natural language, interpretation of a user's query is usually not straightforward. The ambiguity can be either syntactic, e.g., prepositional phrase (PP) attachment, or it can be semantic. In order to resolve such ambiguities, NL enabled question answering systems mostly use general purpose NL parsers. Although these parsers give syntactically correct chunks for a sentence, these chunks might not be semantically meaningful in a domain. For example, consider the following queries:

- "`List the employees working in loss making projects`". In this query, a human can easily disambiguate that "`loss making`" is a modifier of "`projects`", and "`working in loss making projects`" is a modifier of "`the employees`". That is,

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 105-116

Received 23/11/09
Accepted 16/01/10
Final version 10/03/10

the correct chunks are "[List [[the employees] [working [in [[loss making [projects]]]]]]]". However, the chunks obtained from an NL parser are "[List [[[the employees] [working [in [loss]]]] [making [projects]]]]", which will be interpreted as "List the employees who are working in loss and who make projects".

- "Give the projects having costing and billing >$25000 and <$35000, respectively". The NL chunker may chunk this query as "[Give [[the projects] [having [[costing] and [billing]] [>$25000] and [<$35000]]], respectively". From these chunks it is not possible to identify that "costing >$25,000" and "billing <$35,000" are the two constraints which modify "the projects".

Thus the chunks obtained from such parsers may not be helpful in extracting the answer to the user's query. The problem becomes even more severe in case of complex queries involving multiple constraints and nested sub-questions. Thus the problem at hand is "*How can we automatically enrich the output of a general purpose NL parser with the domain knowledge in order to obtain syntactically as well as semantically valid chunks for the queries in the domain?*"

We put forward an approach to solve the queries in a domain using the domain knowledge and the syntactic structure of the queries. A part of the domain knowledge is represented in the form of domain ontology which is based on semantic web technologies [3]. We define '*constraints*' and '*semantically valid chunks*' for a query. Semantically valid chunks aid in the interpretation and extraction of the answers to the user's queries unambiguously.

2 Related Work

Processing natural language questions to obtain an equivalent structured query has long been an area of research in artificial intelligence [2, 1]. Still, most existing approaches cannot interpret real-life natural language questions properly. Even the few which can do so depend so much on domain-specific rules, that porting to other domains becomes an issue.

Popescu et al. [4] adapts the Charniak parser [5] for domain-specific question answering by extending the training corpus of the parser with a set of 150 hand-tagged domain-specific questions. Further, semantic rules inferred from domain knowledge are used to check and correct preposition attachment and preposition ellipsis errors. START [6] decomposes complex questions syntactically or semantically to obtain sub questions that can be answered from available resources. If these answers are not sufficient to solve the question, semantic information - in the form of rules that map 'key' domain questions to the answers - is used. The main drawback of these approaches is that the creation of domain-specific rules is very resource intensive, and hence restricts portability.

AquaLog [7] tries to transform the NL question to ontology-specific triples using syntactic annotations, semantic terms and relations, and question words to interpret the natural language question. If these cannot resolve the ambiguity in the question, domain ontology and/or WordNet are used to make sense of the input query.

3 The Architecture

NL based Question Answering system requires the queries to be analyzed and chunked in an appropriate manner so as to have correct query generation and answer extraction. An NL query can be viewed as consisting of a set of *unknown predicates* whose values need to be determined based on the *constraints* imposed by the rest of the query. Domain ontology along with a POS tagger is used to identify the constraints in the query. These constraints along with the domain knowledge and the parse structure of the query are used to find the semantically valid chunk set. These chunks are then converted to a formal query language and the answer is retrieved from the ontology. Figure 1 demonstrates the overall architecture of our approach. In this figure, solid arrows represent the process flow for a query, and dashed arrows represent the information flow.

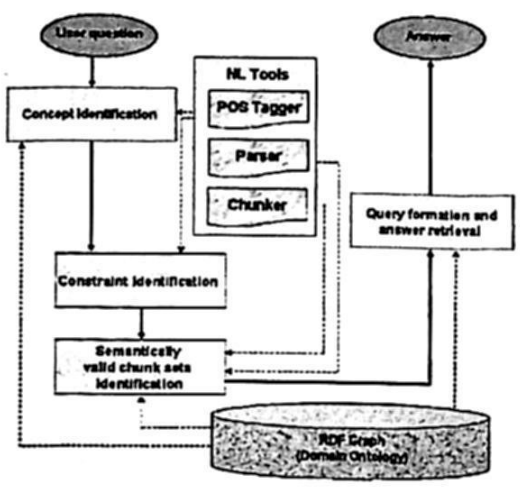

Fig. 1. The Architecture

Hence for the appropriate interpretation and analysis of the query, the important issues that need to be addressed can be summarized as:

Constraint identification. This involves identifying the correct predicate-object pairs.

Semantically valid chunk set. This involves identification of valid constraints for each unknown predicate so that correct interpretation of the given query can be ensured.

Query generation. In this step, the semantically valid chunk set is converted to appropriate formal language query using the domain ontology.

In Section 4 we briefly describe the domain ontology and formalize the terminologies used in this work. In the subsequent sections we discuss constraint identification and semantically valid chunk set formation.

4 Domain Ontology and Other Preliminaries

We use semantic web technologies [3] to create the domain ontology (in RDF format) using the relational data of the business application along with its meta information stored in the seed ontology[1] [8]. The ontology D_O of a domain D describes the domain terms and their relationships in the $\langle subject - predicate - object \rangle$ format. For illustration, $\langle$`Ritesh - project_name - Bechtel`$\rangle$ describes that the predicate '`project_name`' of the subject '`Ritesh`' has object '`Bechtel`'. A synonym dictionary having information about the synonyms of the domain terms is also maintained.

[1] The seed ontology has meta information about the domain, like $\langle ns : employee\ owl : type\ owl : person \rangle$, $\langle ns : employee\ ns : hasName\ ns : employee\_name \rangle$.

The domain ontology and synonym dictionary are used to identify the concepts in the user query Q posed in the domain D. The domain ontology D_O is used to further classify the concepts as predicates and objects. For a query Q, we denote the *set of predicates* as $P_Q = \{p_1, p_2, \ldots p_n \mid \exists\langle s - p_i - o\rangle \in D_O$, and p_i is present in the query $Q\}$. The *set of objects* present in the query Q is $O_Q = \{o_1, o_2, \ldots o_m \mid \exists\langle s - p - o_i\rangle \in D_O$ or o_i is a numerical/date value, and o_i is present in the query $Q\}$.

In this work, we have considered examples from a logical subset of the Project Management System for an organization. The database consists of tables containing data about the projects, the various costs associated with the projects, employees and their allocations in different projects. The important tables are named ProjectDetails, Employees, CostDetails and Allocations. Some of the attributes of these tables are:

ProjectDetails. project_id, project_name, project_type
Employees. employee_id, employee_name, age, gender, joining_date
CostDetails. costDetails_id, project_id, costing, billing, revenue
Allocations. allocation_id, employee_id, project_id, role

For illustration, consider the query:

Example 1. `What is the role of the associates who joined before 18/10/2008 in the SWON projects with costing and revenue more than $15000 and < $25000, respectively?`

In this query, the concepts identified using the domain ontology are: `role`, `employee_name`[2], `joining_date`, `SWON`, `project_name`, `costing`, and `revenue`. After considering the date and the numeric values in the query, i.e., `18/10/2008`, `15000` and `25000` as objects, the predicate and object set obtained are:

$P_Q = \{$`role, employee_name, joining_date, project_name, costing, revenue`$\}$,
$O_Q = \{$`SWON, 18/10/2008, 15000, 25000`$\}$.

In the following section we present an algorithm to bind these predicates and objects so as to obtain constraints for the query.

5 Constraint Identification

For a successful query creation and execution, identification and formulation of correct constraints is of utmost importance. With reference to this work, constraint identification involves binding each '*object*' in the query with its corresponding '*predicate*'. This predicate-object pair is referred to as 'constraint'.

We define a *constraint* as $c_k = (p_i, o_j), o_j \in O_Q, p_i \in P_Q$, and o_j is the value for the predicate p_i in Q. All the constraints in the query Q are identified, and $C_Q = \{c_1, c_2, \ldots c_m\}$ denotes the *constraints set.* Predicate used in any constraint is referred to as *constraint predicate.* The *set of constraint predicate* is $P_Q^C = \{p_i \mid p_i \in P_Q$ such that $\exists(p_i, o_i) \in C_Q\}$. Predicates that do not form part of the constraint set are referred to as *unknown predicates.* The *set of unknown predicates* is $P_Q^U = \{p_i \mid p_i \in P_Q, p_i \notin P_Q^C\}$.

[2] The synonym dictionary has mappings defined between '`employee`' and '`associate`' etc.

In a natural language query, constraint identification (or predicate-object binding) needs special attention due to the following reasons:

Unspecified predicates. For some (or all) of the objects present in the query, the corresponding predicate might not be explicitly specified. For example consider the query "`Give me the role of Puneet in the project having Ritesh as project leader`". Here the objects '`Puneet`' and '`Ritesh`' need to be attached to the corresponding predicate '`employee_name`', which is not specified in the query.

Constraint vs. unknown predicate. The issue of unspecified predicates becomes even more severe when a predicate p_i for an object o is present in the query, but the same predicate p_i also happens to be an unknown predicate. For example, in the query mentioned above, the value '`project leader`' in O_Q is compatible to the predicate '`role`' in P_Q. But these predicate and value are not to be bound as the predicate '`role`' is an unknown predicate, whose value needs to be determined.

Predicates followed by the respective objects. In questions with multiple constraints, sometimes predicate and its object may not be given consecutively. Instead, the query may have a predicate list followed by the corresponding object list (or vice versa). Considering the same Example 1 (Section 4), in the phrase '`with costing and revenue more than $15000 and < $25000, respectively`', the predicate list, i.e. [`costing, revenue`], is followed by the corresponding objects list, i.e. [`more than $15000, < $25000`]. We need to identify and bind the appropriate predicate-operator-object pairs from the predicate and the object lists.

In the following we describe an algorithm for predicate-object discovery and binding.

5.1 Algorithm for Predicate-Object Binding

The main steps of the algorithm are as follows.

Step 1. Operator-Object binding for numerical/date objects. The first step towards Operator-Object binding is the identification of the comparison operators in the query. For operator identification, the system maintains a mathematical operator dictionary. All the occurrences of the string comparators in the question are replaced by the corresponding mathematical comparator. Also, if there is any numeric value in the question that is not preceded by any operator, by default '=' operator is prefixed. These form the corresponding operator-object pairs. Henceforth we refer to these operator-object pairs as objects.

Step 2. Group the predicates and objects that immediately follow/preceed the POS tags of the *assignment* words[3]. POS tags of some such words are '`VBZ`', '`VBP`', '`IN`', '`SYM`' etc. We also group the predicates that are immediately followed (or preceded) by any object. In case there is a list of predicates and a list of objects satisfying the above, then these lists are also grouped. These groups are the *possible pairs* for predicate-object binding. For instance, in Example 1, we

[3] Assignment words are words which are usually specified between the predicates and its objects. E.g., form of copula 'be', preposition 'as', or mathematical operators.

get the pairs (joining_date : <18/10/2008), (project_name, (SWON)), (costing, revenue : >$15000, <$25000).

Step 3. From the groups obtained in Step 2, we bind the predicates and objects that are of the same data type. The compatibility for predicate and the object is checked using the domain ontology. In case of predicate and object list, one-on-one binding is done. For example, from the groups obtained above, we get the following predicate-object pairs: (joining_date, <18/10/2008), (costing, >$15000), and (revenue, <$25000). The pair (project_name, SWON) is not bound because the predicate of the object 'SWON' as obtained from the domain ontology is 'project_type'.

Step 4. The string objects that are not bound to any predicate in Step 3 are bound to their compatible predicates. The compatible predicate for an object is determined using the domain ontology. For instance, since the object 'SWON' is not bound to any predicate till now, it is bound to its compatible predicate 'project_type' to obtain the constraint (project_type, SWON).

The predicates bound to any object in the above steps form the constraint predicate set, and the remaining predicates constitute the unknown predicate set. Using the above algorithm for Example 1, we obtain the constraint set, constraint predicate set and the unknown predicate set as:

- C_Q = {(joining_date, <18/10/2008), (project_type, SWON), (costing, >$15000), (revenue, <$25000)}.
- P_Q^C = {joining_date, project_type, costing, revenue}.
- P_Q^U = {role, employee_name, project_name}.

The constraint sets thus obtained are used to find the semantically valid chunk set as discussed in the following section.

6 Semantically Valid Chunk Set

Semantically valid chunk set identify the conditions on each unknown predicate in the query, and are constituted from the constraints and unknown predicates as obtained in Section 5. For instance, in Example 1 (Section 4), 'joining_date < 18/10/2008' is a condition on the predicate 'employee_name'. Due to the syntactic ambiguity, more than one syntactic parse might be obtained for a NL query. Such cases may eventually result in more than one semantically viable chunk set. Formally we define semantically viable chunk sets as follows.

Definition. A *Semantically viable chunk set* (SVC set) of a query Q corresponding to the k^{th} parse is a set $SVC_{Q_k} = \{SC^p_{Q_k} \mid p \in P^U_Q\}$ where $SC^p_{Q_k}$ is a semantic chunk. *Semantic chunk* of a predicate $p \in P^U_Q$ is defined as:

- If $C_Q \neq \{\}$, $SC^p_{Q_k} = \langle p, c_1, c_2, \ldots c_i, \ldots c_r \rangle$ $(r \geq 1)$, where $c_i \in C_Q$ or $c_i = SC^{p'}_{Q_k} \in SVC_{Q_k}$, and c_i is a condition on the predicate p
- If $C_Q = \{\}$ (and $P^U_Q \neq \{\}$), $SC^p_{Q_k} = \langle p \rangle$

Such that, SVC_{Q_k} satisfies the following:

a. $\forall p \in P_Q^U, \exists SC_{Q_k}^p \in SVC_{Q_k}$.
b. $\forall c' \in C_Q, \exists SC_{Q_k}^p \in SVC_{Q_k}$ such that $SC_{Q_k}^p = (p, c_1, c_2, \ldots c', \ldots c_r)$.

The condition 'a' states that there is a semantic chunk for each unknown predicates in the query. The condition 'b' states that each constraint in the query is used in at least one semantic chunk.

Definition. For a query Q, the semantically viable chunk set which is semantically valid as per the domain ontology is the *semantically valid chunk set*, $SVaC_Q$. These sets are referred to as SVaC sets.

We can classify the queries posed for a natural language interface to business application in the following categories:

1. The unknown predicate is specified as a noun in the question. E.g., '`What is the role of Ritesh in AB Corp?`', '`Give me the project of Ritesh`'. In this case the syntactic modifiers of the predicate (noun) determine the constraints on the predicate.
2. A wh-word ('`who/when/where`') refers to the unknown predicate. E.g., '`Who is the project leader of Bechtel?`', '`When did Ritesh join Bechtel?`'
3. The unknown predicate may have a wh-word ('`what/which/whose/how much/how many`') as a determiner. In such questions the wh-word is placed before the unknown predicate, and these words ask which thing or person is being referred to. For example, '`In which project is Ritesh allocated?`', '`How many employees are allocated to Bechtel?`'
4. `Why/how` questions. These questions expect descriptive answer. Since our system aims to handle only factual questions, such questions are out of scope.
5. Yes/No questions. These questions expect '`yes`' or '`no`' as answer. Due to space limitations, such questions are kept out of scope of the current work.

We use syntactic information of the question to obtain the semantically viable chunk sets as discussed in the following section.

6.1 Finding Semantically Viable Chunk Sets

For a query, the main task for identification of semantically viable chunk sets is to identify the conditions for all the unknown predicates. We exploit syntactic information of the query for this purpose. We use a dependency-based parser (e.g. Stanford Parser [9], Link Parser [10]) to obtain the syntactic structure of the question. These parsers provide the phrase structure as well as the dependencies between different words of a given sentence. In case of syntactic ambiguity, these parsers provide all possible interpretations of the input sentence. In the following, we discuss the process of identifying the appropriate semantic chunks for different categories of queries.

Unknown Predicate as Noun. If an unknown predicate in the query plays the role of noun, its syntactic modifiers identify the constraints on the predicate. Dependency based parsers provide dependencies between noun and its modifiers. This information along with the phrase structure of the query is

used to determine the phrase modifying the unknown predicate. These phrases give the constraints for the unknown predicate. The unknown predicate with its constraint is a candidate semantic chunk. For example, for the question 'Give me the role of the associates with age > 30 years?', the preposition phrase 'with age > 30 years' is a post-nominal modifier of the noun 'associates'. The constraint corresponding to this preposition phrase is 'age > 30', and hence the corresponding semantic chunk can be obtained as

$SC_Q^{employee\_name} = \langle employee\_name, age > 30 \rangle$.

Further, the preposition phrase 'of the associates with age > 30 years' is modifying the noun 'role'. Since the semantic chunk, $SC_Q^{employee\_name}$, for the phrase 'of the associates with age > 30 years' has already been identified, the semantic chunk for the predicate 'role' is $SC_Q^{role} = \langle role, SC_Q^{employee\_name} \rangle$.

Unknown Predicate as wh-word. In a domain 'who' usually refers to a person, such as 'employee_name', 'student_name'; 'when' refers to date/time attributes like 'joining_date', 'completion_time'; and 'where' refers to locations like 'address', 'city'. For the given business application, this information about the wh-words is identified, and stored in the seed ontology. In questions involving any of these wh-word, the predicate corresponding to the wh-word is found using the domain ontology, which might be a possible candidate for being a unknown predicate. If the wh-word in the question is compatible to more than one predicate in the domain, then more semantic chunks - corresponding to each compatible predicate - are obtained. Semantic information is used in such cases to resolve the ambiguity regarding the most appropriate predicate (See Section 6.2). The constraints of the wh-word are determined on the basis of the role of the wh-word in the question as discussed below.

- If the wh-word is the subject in the question, the corresponding verb phrase determines the constraint on the wh-word. For example, consider the query 'Who is the project leader of Bechtel?' In this question, the verb phrase 'is the project leader of Bechtel' gives the constraints of 'who'. The constraints embedded in this verb phrase are 'role=project leader' and 'project_name=Bechtel'. Since in our domain 'who' corresponds only to the predicate 'employee_name', the semantic chunk is $\langle employee\_name,\ role = project\ leader,\ project\_name = Bechtel \rangle$.
- In other cases, the words in the phrase enclosing the wh-word determines the constraints on the wh-word. For example, for the question 'When did Ritesh join Bechtel?', the chunk structure as given by a NL parser is '[S When did [NP Ritesh NP] [VP join [NP Bechtel NP] VP] S]'. In our domain 'when' corresponds to 'joining_date'. The constraints in this question are 'employee_name=Ritesh' and 'project_name=Bechtel'. Thus, the semantic chunk obtained is $\langle joining\_date,\ employee\_name = Ritesh,\ project\_name = Bechtel \rangle$.

Wh-word as the Determiner of the Unknown Predicate. In this case also the constraints are determined as discussed above. For example, consider the question 'In which project is Ritesh allocated?'. The constraint for the unknown predicate 'project_name' can be identified as 'employee_name=Ritesh'. Thus the semantic chunk is $\langle project\_name, employee\_name = Ritesh \rangle$.

Using the syntactic information as discussed above, all possible semantic chunks for a parse structure of the question are determined. The set of these chunks is a semantically viable chunk set only if the chunk set satisfies the conditions (a) and (b) specified in the definition of SVC sets. For instance, for the query in Example 1, two SVC sets are obtained as given in Figure 2.

$SVC_{Q_1} = \{SC_{Q_1}^{role}, SC_{Q_1}^{employee\_name}, SC_{Q_1}^{project\_name}\}$, where:
- $SC_{Q_1}^{project\_name}$ = (*project_name*, *project_type* = *SWON*, *costing* > $15000, *revenue* < $25000);
- $SC_{Q_1}^{employee\_name}$ = (*employee_name*, *joining_date* < 18/10/2008);
- $SC_{Q_1}^{role}$ = (*role*, $SC_{Q_1}^{project\_name}$, $SC_{Q_1}^{employee\_name}$).

And,
$SVC_{Q_2} = \{SC_{Q_2}^{role}, SC_{Q_2}^{employee\_name}, SC_{Q_2}^{project\_name}\}$, where:
- $SC_{Q_2}^{project\_name}$ = (*project_name*, *project_type* = *SWON*);
- $SC_{Q_2}^{employee\_name}$ = (*employee_name*, *joining_date* < 18/10/2008, *costing* > $15000, *revenue* < $25000);
- $SC_{Q_2}^{role}$ = (*role*, $SC_{Q_2}^{project\_name}$, $SC_{Q_2}^{employee\_name}$).

Fig. 2. SVC Sets for Example 1

If for a query Q, only one semantically viable chunk set is found then this chunk set is the semantically valid chunk set. In other cases, the semantically valid chunk set is found by using the domain specific semantic information as discussed in the following section.

6.2 Finding Semantically Valid Chunk Sets

If more than one semantically viable chunk sets are obtained for a question, semantic information obtained from the domain ontology is used to determine the semantically valid chunk set. Let $SVC_{Q_1} = \{SC_{Q_1}^{p} | p \in P_Q^U\}$ and $SVC_{Q_2} = \{SC_{Q_2}^{p} | p \in P_Q^U\}$ be any two SVC sets for a query Q. Since there are more than one SVC set for Q, $\exists p_i, p_j \in P_Q^U$, and $c' = (p', v') \in C_Q$ such that c' is a constituent of $SC_{Q_1}^{p_i} \in SVC_{Q_1}$ and $SC_{Q_2}^{p_j} \in SVC_{Q_2}$. But, in the valid interpretation of Q, c' can specify either the unknown predicate p_i or the unknown predicate p_j. Hence we conclude that, in this case, the syntactic information is not sufficient to resolve the ambiguity whether c' is a constraint of p_i or p_j.

For instance, consider the SVC sets of the query in Example 1 (Figure 2). In the two SVC sets obtained for this query, the constraints '`costing > $15000`' and '`revenue < $25000`' are bound to '`project_name`' in SVC_{Q_1}, and to '`employee_name`' in SVC_{Q_2}.

To resolve such ambiguities, we use *depth* between the concerned predicates. The number of tables required to be traversed[4] in order to find relationship between any two predicates is determined through the domain ontology. This is referred to as the depth between the two predicates. If for a pair of predicates,

[4] This is found using the primary and foreign key information of the tables.

there exists more than one path then we choose the one with the minimum depth. It is observed that the semantic chunk in which the unknown predicate and the constraint predicate pair has lesser depth is the one which is more likely to be the correct pair. We use domain ontology to find the depth between two predicates as described below.

Step 1. Breadth first search (BFS): The system does a BFS on the tables in the ontology to determine if p_i or p_j belongs in the same table as that of p'. Without loss of generality, assume that p_i and p' belong to the same table, and p_j does not belong to the table of p'. In this case, $SC_{Q_1}^{p_i}$, and consequently SVC_{Q_1} is assumed to be correct, and SVC_{Q_2} is rejected. This in this case, $SVaC_Q = SVC_{Q_1}$.

Step 2. Depth first search (DFS): We involve DFS method to resolve the ambiguity regarding the constraint c' if BFS is not able to do so. The depth of the path from p' to p_i and p_j is found using the domain ontology. The constraint c' is attached to the predicate with which the distance of p' is minimum, and the corresponding SVC set is the semantically viable chunk set.

In Example 1 (Section 4), the constraint predicates '`revenue`' and '`costing`' are found to be closer to the predicate '`project_name`' than to the predicate '`employee_name`'. Hence, the semantically viable chunk set SVC_{Q_1} is the semantically valid chunk set.

An advantage of this approach is that depending upon the question complexity the system does a deeper analysis. Domain ontology is used only if a question cannot be resolved by using just the syntactic information. If domain information also is not sufficient for question interpretation, then answers for all interpretations are found, and the user is asked to choose the correct answer.

7 Formal Query Formation

The semantic chunks of the SVaC set are processed by the QueryManager. In this module, a formal query is generated on-the-fly from the semantic chunks to extract the answer of the user's question. Since the domain ontology is in RDF format, we generate queries in SPARQL[5] which is a query language for RDF.

For a semantically valid chunk set, we start with formulating SPARQL queries for the semantic chunks which do not contain any sub-chunk. The unknown predicate of the semantic chunk forms the 'SELECT' clause, and the constraints form a part of the 'WHERE' clause.

For instance, from the SVaC set $SVaC_Q$ (i.e., SVC_{Q_1} in Figure 2) of Example 1, we first formulate SPARQL query for the semantic chunks $SC_{Q_1}^{project\_name}$ and $SC_{Q_1}^{employee\_name}$, and obtain the answer from the RDF. Assume that the answers of these chunks are obtained as "[`Quantas`, `Bechtel`, `NYK`]", and "[`Rajat`, `Nidhi`]". The answers obtained from independent semantic chunks are then substituted in the semantic chunks involving nested sub-chunks. For example, to formulate the query for the semantic chunk $SC_{Q_1}^{role}$, the answers of the semantic

[5] http://dev.w3.org/cvsweb/2004/PythonLib-IH/Doc/sparqlDesc.html?rev=1.11

chunks $SC_{Q_1}^{project\_name}$ and $SC_{Q_1}^{employee\_name}$ are used. Therefore, upon substituting these answers, the semantic chunk $SC_{Q_1}^{role}$ is modified as:
$SC_{Q_1}^{role} = \{role, project\_name = Quantas, project\_name = Bechtel, project\_name = NYK, employee\_name = Rajat, employee\_name = Nidhi\}$. The SPARQL query for this chunk is then generated, and answer of the user query is retrieved.

8 Experimental Results

To test the approach discussed above, we carried out experiments on various domains (project management, retail, asset management). Users were asked to pose queries to an existing question answering system [8]. A set of almost 2750 questions were posed to the system, out of which approximately 35% consisted of fairly simple questions (No chunking required, but predicate-value binding required) e.g. "`List all projects with costing less than $30000`". The remaining 65% questions required correct predicate-value binding as well as correct chunking, like "`list the employees in the projects having costing less than $30000, started on 2009-10-10 with Ritesh as group leader`". When compared with actual answers following observations were made:

- 791 out of 946 questions were answered correctly for simple questions, which sums up approximately to 83.6%.
- For complex queries, 431 out of 1804 were correctly answered which approximately accounts 23.9%

The users' questions were again tested on the new system as discussed in this work. We have used Link Parser [10] for the syntactic analysis of the queries. The link parser yields the syntactic relations between different words of the sentence in the form of labeled links. In case of ambiguity in the input sentence, the link parser yields syntactic structures corresponding to all possible interpretations.

The following observations were made:

- 863 out of 946 questions were answered correctly for simple questions, which sums up approximately to 91.2%.
- 1508 out of 1804 were correct from complex questions that accounts 83.6%

Comparing the results of the two approaches, a direct increase of about 7% was attained for simple questions and for complex queries, it went up by almost 58%. The increment in the correctness of the answers were due to the following:

- Due to the predicate object binding, correct bindings were obtained (even for simple questions) which increased the number of answers correctly.
- The chunks obtained from the link parser when furnished with domain knowledge in the form of constraints, helped in achieving higher correct results.

Approximately 13% of the questions were not answered at all, or were answered incorrectly, or in some cases partially correct answers were obtained. After the analysis as to why the answers to those queries were not obtained, following conclusions were made.

- Syntactic analysis by the parser was not correct.
- The question posed by the user was grammatically incorrect.
- The system was not able to capture the semantic knowledge. E.g. for the queries '`Who is reporting to Ritesh?`' and '`Who Ritesh is reporting to?`", the system could not fetch correct answers.

We are currently working towards handling these issues.

9 Conclusion

We have described an approach to obtain semantically valid chunk set for NL-enabled business applications. For any question posed by a user to a business application system in natural language, the system should be robust enough to analyze, understand and comprehend the question and come up with the appropriate answer. This requires correct parsing, chunking, constraints formulation and sub-query generation. Although most general purpose parsers parse the query correctly, due to lack of domain knowledge, domain relevant chunks are not obtained. Therefore, in our work we have concentrated on enriching general purpose parsers with domain knowledge using domain ontology in the form of RDF. We have handled constraints formulation and sub query generation which form the backbone of any robust NL system. Tackling all these issues make any natural language-enabled business application system more robust, and enables it to handle even complex queries easily, efficiently and effectively.

References

1. Lopez, V., Motta, E., Uren, V., Sabou, M.: State of the art on semantic question answering - a literature review. Technical report, KMI (May 2007)
2. Androutsopoulos, I., Ritchie, G., Thanisch, P.: Natural language interfaces to databases - an introduction. Natural Language Engineering 1(1) (1995) 29–81
3. Antoniou, G., van Harmelen, F.: A Semantic Web Primer. The MIT Press (2004)
4. Popescu, A.M., Armanasu, A., Etzioni, O., Ko, D., Yates, A.: Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In: 20th international conference on Computational Linguistics, Geneva, Switzerland (2004)
5. Charniak, E.: A maximum-entropy-inspired parser. In: NAACL. (2000)
6. Katz, B., Borchardt, G., Felshinm, S.: Syntactic and semantic decomposition strategies for question answering from multiple resources. In: AAAI 2005 Workshop on Inference for Textual Question Answering, Pittsburgh, PA (July 2005) 35–41
7. Lopez, V., Motta, E., Uren, V.: Aqualog: an ontology-driven question answering system to interface the semantic web. In: NAACL on Human Language Technology, New York (2006) 269 – 272
8. Bhat, S., Anantaram, C., Jain, H.K.: A framework for intelligent conversational email interface to business applications. In: ICCIT, Korea (2007)
9. Klein, D., Manning, C.D.: Fast exact inference with a factored model for natural language parsing. In: Advances in Neural Information Processing Systems. Volume 15., Cambridge, MA, MIT Press (2003) 3–10
10. Grinberg, D., Lafferty, J., Sleator, D.: A robust parsing algorithm for link grammars. In: Fourth International Workshop on Parsing Technologies, Prague (September 1995)

Opinion, Emotions, Textual Entailment

Word Sense Disambiguation in Opinion Mining: Pros and Cons

Tamara Martín-Wanton[1], Alexandra Balahur-Dobrescu[2], Andrés Montoyo-Guijarro[2] and Aurora Pons-Porrata[1]

[1]Universidad de Oriente, Center for Pattern Recognition and Data Mining
Patricio Lumumba s/n, Santiago de Cuba, Cuba
{tamara, aurora}@cerpamid.co.cu
[2]University of Alicante, Departament of Software and Computing Systems,
Apartado de Correos 99, E-03080 Alicante, Spain
{abalahur, montoyo}@dlsi.ua.es

Abstract. The past years have marked the birth of a new type of society - that of interaction and subjective communication, using the mechanisms of the Social Web. As a response to the growth in subjective information, a new task was defined - opinion mining, dealing with its automatic treatment. As the majority of natural language processing tasks, opinion mining is faced with the issue of language ambiguity, as different senses of the same word may have different polarities. This article studies the influence of applying word sense disambiguation (WSD) within the task of opinion mining, evaluating the advantages and disadvantages of the approach. We evaluate the WSD-based method on a corpus of newspaper quotations and compare it to the results of an opinion mining system without WSD. Finally, we discuss our findings and show how WSD helps in the task of opinion mining.

1 Introduction

The past years, with the growing volumen of subjective data originating from texts pertaining to the Social Web -blogs, reviews, forums, discussion panels - have marked the birth of a new type of society, where individuals can freely communicate and exchange opinions. The large benefits that can be obtained by the analysis of this data (more informed customers, companies, societies) made essential the study of methods that can be automatically employed to extract the required information.

Therefore, over the past few years, there has been a large increase of interest in the identification and automatic extraction of the attitudes, opinions and feelings expressed in texts. This movement is given by to the need to provide tools for users of different domains, which require, for different reasons, the automatic monitoring of information that expresses opinion. A system that automatically carries out this task would eliminate the effort to manually extract useful knowledge from the information available on the Web.

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 119-129

Received 30/11/09
Accepted 16/01/10
Final version 08/03/10

Opinion Mining (also known as sentiment classification or subjectivity analysis) covers a wide area of Natural Language Processing, Computational Linguistics and Text Mining. The goal is not to determine at the topic of a document is, but the opinion that is expressed in it. Therefore, its objective is to determine the opinion of a speaker or a writer on a topic [1].

Many approaches to sentiment analysis rely on lexicons of words that may be used to express subjectivity; these works do not make distinction between different senses of a word, so that the term, and not its senses, are classified. Moreover, most subjectivity lexicons are compiled as lists of keywords, rather than word meanings. However, many keywords have both subjective and objective senses and even the purely subjective senses have a degree of positivity or negativity, depending on the context where the corresponding word appears.

This paper presents a research on the advantages of using the word sense disambiguation to determine the polarity of opinions and the role of existing resources. To this end, we evaluated two unsupervised approaches, one based on lists of positive and negative words and the other based on a word sense disambiguation algorithm over the same affect lexicons.

2 Related Work

A major task of Opinion Mining consists in classifying the polarity of the extracted opinion. This process determines whether the opinion is positive, negative or neutral with respect to the entity which it is referring to (for example, a person, a product, a topic, a movie, etc.).

The large majority of research in this field focused on annotating the sentiment of the opinions but out of context (e.g. [2–5]). Recently some works have been published determining the polarity of the word senses, thus building resources that can be useful in different tasks of Opinion Mining ([1, 6, 7]). Esuli and Sebastiani [1] determine the polarity of word senses in WordNet, distinguishing among positive, negative and objective. They manually annotate a seed set of positive/negative senses in WordNet and by following the relations in WordNet expand the small set using a supervised approach. They extend their work [6] by applying the Page Rank algorithm for ranking the WordNet senses in terms of how strongly a sense possesses a given semantic property (e.g., positive or negative). Wiebe and Mihalcea [7] label word senses in WordNet as subjective or objective. They use a method relying on distributional similarity as well as an independent, large manually annotated opinion corpus (MPQA) [8] for determining subjectivity.

Only few recent works take into account the correct senses of the words in the opinions (e.g. [9, 10]), but they are supervised methods. Akkaya et al. [9] build and evaluate a supervised disambiguation system that determines whether a word in a given context is being used with objective or subjective senses. In this approach the inventory of objective and subjective senses of a word can be viewed as an inventory of the senses of the word but with a coarse granularity. Rentoumi et al. [10] go a step further and determine the polarity

by disambiguating the words and then mapping the senses to models of positive and negative polarity. To compute these models and produce the mappings of senses, they adopt a graph-based method which takes into account contextual and sub-word information.

3 Motivation and Contribution

Whereas most research concentrates on the analysis of opinions at a word level there are some approaches dealing with the analysis at the sense level. The reason for this fact is that the meaning of most words depends on the context where they appears and in order to determine the polarity of opinions, it is also important to take into account the meanings of the words and the relations between them. The recent studies that regard word senses concentrate on the detection of subjectivity or on ranking senses in a lexicon according to their polarity, but they do not have as main aim the classification of the polarity. On the other hand, most research concentrates on assessing the polarity of opinion using one of the available lexicons of opinion. However, using a resource or another cannot give a measure of the impact the use of these resources has on the final system results.

The main motivation of the present research is to study the impact of word sense disambiguation for determining the polarity of opinions. There are approaches that perform the analysis at the sense level, but they lack of a thorough study of the advantages and disadvantages of the disambiguation as intermediate task. They do not perform the analysis of the approaches at a word and sense levels on the same corpus and using the same resource.

Thus, the contribution of this paper is given by the evaluation of two unsupervised approaches on the same corpus and using the same resources, both are based on knowledge but differ in the level at which the analysis is performed (word and sense level). The second objective is to carry out an analysis of existing public resources for opinion mining and their influence on both approaches. In this way, we can have a clear measure of the impact given by the use of each of the resources separately, as well as study methods to combine them, to obtain better results.

We show that word sense disambiguation avoids the lack of balance between the classification of positive and negative opinions present in an approach as simple as positive or negative words belonging to an opinion. The majority of the existing resources that have annotated senses with its corresponding polarity have little coverage.

4 Experiments and Evaluation

In the experiments, we intend to evaluate the impact of the disambiguation of words in the task of polarity classification of an opinion. With this aim, we first present a "bag of words" approach, and then a second one that uses a word sense disambiguation algorithm to determine the correct sense of the words in

the opinion. In both approaches, we firstly perform a pre-processing of the text including sentence recognition, stop-word removal, part-of-speech tagging and word stemming by using the TreeTagger tool [11].

We comparatively analyse the different possible methods and resources for opinion mining that are publicly available and explore the possibility to combine them in order to increase the accuracy of the classification. For the evaluation of both methods we use precision, recall and F1 measures for the Positive (P+, R+, F1+) and Negative (P-, R-, F1-) categories, and the overall precision, recall and F1 (P, R, F1). Additionally, the coverage (Cov) of the method is calculated as the ratio of the number of opinions classified as negative or positive over the total number of opinions.

4.1 Data and Resources

For our experiments, we chose a set of 99 quotes described in [12], on which agreement between a minimum of two annotators could be reached regarding their classification in the positive and negative categories, as well as their being neutral/controversial or improperly extracted. In this paper, we only use the 68 quotes classified as positive (35) or negative (33). The explanation for employing this dataset is that reported speech (a person referring to another person or event) represents a direct and unbiased expression of opinions that does not depend on the interpretation of the reader in the majority of cases.

At the present moment, there are some lexicons annotated with affect and polarity at the sense level and are based on WordNet [13]: WordNet-Affect [14], SentiWordNet [1] and Micro-WNOp [15]. WordNet-Affect, an extension of WordNet Domains, is a hierarchy of affective domain labels which were developed by selecting suitable synsets from WordNet which represent affective concepts and dividing them into subsets of affective data. In SentiWordNet, each synset in WordNet has assigned three values of sentiment: positive, negative and objective, whose sum is 1. For example, the synset HAPPY#3 (marked by good fortune; "a felicitous life"; "a happy outcome"), is annotated as Positive = 0.875, Negative = 0.0 and Objective = 0.125. This resource was created through a mix of linguistic techniques and statistical classifiers. It was semi-automatically built so all the results were not manually validated and some resulting classifications can appear incorrect. Finally, the Micro-WNOp corpus is composed by 1105 WordNet synsets manually annotated in a manner that is similar to SentiWordNet.

4.2 Word-based Method Applied to Polarity Classication

For the first approach, as in [12], each of the employed resources were mapped to four categories, which were given different scores - positive (1), high positive (4), negative (-1) and high negative (-4). On the one hand, the approach is motivated by the same mapping done in [12], and, on the other, on the wish to mantain the "idea" of the lexicons employed - that the same term may have different strengths of polarity and that two different terms even if they have the same polarity, may differ in intensity.

The words belonging to the WordNet-Affect categories of anger and disgust were grouped as in [12] under high negative, fear and sadness were considered negative, joy was taken as containing positive words and surprise as highly positive; SentiWordNet and Micro-WNOp contain positive and negative scores between 0 and 1 and in their case, the mapping was done in the following manner: the words that have senses with positive scores lower than or equal to 0.5 to the positive category, the scores higher than 0.5 to the high positive set, the negative scores lower than or equal to 0.5 to the negative category and the ones higher than 0.5 to the high negative set. See Table 1 for the statistics of the categories built from each resource. The last row corresponds to the union of the categories for all resources.

Table 1. Statistics of the categories used by the word-based method.

Resource	Positive	Negative	High Positive	High Negative
WN-Affect	192	215	73	201
Micro-WNOp	436	396	409	457
SentiWN	23133	22144	2462	5279
SentiWN+Micro-WNOp+WN-Affect	23394	22442	2804	5713

Finally, the polarity value of each of the quotes was computed as sum of the values of the words identified; a positive score leads to the classification of the quote as positive, whereas a final negative score leads to the system classifying the quote as negative. A quote is classified as neutral if the score is equal to 0. Note that no word sense disambiguation is done in this method, rather a word is incorporated into a category depending on the annotation of its senses. Thus, a same word can be included to several categories and the word-to-sense relationship is lost. The results of this approach are shown in Table 2.

Table 2. Classification results of the method without WSD.

Resources	P+	P-	R+	R-	F1+	F1-	P	R	F1	Cov
WN-Affect	0.75	0.60	0.08	0.09	0.15	0.16	0.67	0.09	0.15	0.13
Micro-WNOp	0.57	0.67	0.57	0.18	0.57	0.28	0.59	0.38	0.46	0.65
SentiWN	0.55	0.46	0.48	0.36	0.51	0.41	0.51	0.43	0.46	0.84
SentiWN+WN-Affect	0.55	0.46	0.48	0.36	0.51	0.41	0.51	0.43	0.46	0.84
SentiWN + Micro-WNOp	0.53	0.48	0.54	0.33	0.54	0.39	0.51	0.44	0.47	0.87
All	0.53	0.45	0.54	0.33	0.53	0.38	0.50	0.44	0.46	0.88

4.3 WSD Method Applied to polarity classification

This approach based on word sense disambiguation to determine the polarity of opinions was previously presented in [16], where it was evaluated over the SemEval Task No. 14: Affective Text data, outperforming the results obtained by both unsupervised and supervised systems participating in the competition.

Word Sense Disambiguation (WSD) is an intermediate task of Natural Language Processing. It consists in selecting the appropriate meaning of a word given the context in which it occurs [17].

The approach is based on the assumption that the same word, in different contexts, may not have the same polarity. For example, the word "drug" can be positive, negative or objective, depending on the context where it appears (e.g., "she takes drugs for her heart" (objective), "to be on drugs" (negative)). Bearing in mind this need to appropiately identify the correct sense, we use a word sense disambiguation algorithm to obtain the correct sense of the words in the opinion and subsequently obtain the polarity of the senses from resources based on senses annotated with valence and emotions. The WSD-based method also handles negations and other polarity shifters obtained from the General Inquirer dictionary.

For the disambiguation of the words, we use the method proposed in [18], which relies on clustering as a way of identifying semantically related word senses. In this WSD method, the senses are represented as signatures built from the repository of concepts of WordNet. The disambiguation process starts from a clustering distribution of all possible senses of the ambiguous words by applying the Extended Star clustering algorithm [19]. Such a clustering tries to identify cohesive groups of word senses, which are assumed to represent different meanings for the set of words. Subsequently, clusters that best match the context are selected. If the selected clusters disambiguate all words, the process stops and the senses belonging to the selected clusters are interpreted as the disambiguating ones. Otherwise, the clustering process is performed again (regarding the remaining senses), until a complete disambiguation is achieved.

Once the correct sense for each word on the opinion is obtained, the method determines its polarity regarding the sentiment annotation for this sense in the lexical resource utilized. From SentiWordNet and Micro-WNOp we obtain a positive and a negative value for the target sense (in Micro-WNOp only a part of the synsets are annotated with the polarity, thus the senses that are not annotated are considered to be completely objectives). In the case of WordNet-Affect that is annotated with emotions and not with values of polarity as such, we build a mapping, the senses pertaining to the hierarchy of positive (negative) affective domain labels were assigned a positive value of 1(0) and a negative value of 0(1), respectively.

Finally, the polarity of the opinion is determined from the scores of positive and negative words it contains. To sum up, for each word w and its correct sense s, the positive ($P(w)$) and negative ($N(w)$) scores are calculated as:

$$P(w) = \text{Positive value of } s \text{ in a } \textit{lexical resource}$$

$N(w)$ = Negative value of s in a *lexical resource*

Finally, the global positive and negative scores (S_p, S_n) are calculated as:

$$S_p = \sum_{w:P(w)>N(w)} P(w)$$

$$S_n = \sum_{w:N(w)>P(w)} N(w)$$

If S_p is greater than S_n then the opinion is considered as positive. On the contrary, if S_p is less than S_n the opinion is negative. Finally, if S_p is equal to S_n the opinion is considered as neutral. In Table 3 the results are shown.

Table 3. Classification results of the WSD-based method.

Resources	P+	P-	R+	R-	F1+	F1-	P	R	F1	Cov
WN-Affect	1.00	0.75	0.17	0.09	0.29	0.16	0.90	0.13	0.23	0.15
Micro-WNOp	0.58	0.40	0.20	0.06	0.30	0.11	0.53	0.13	0.21	0.25
SentiWN	0.48	0.53	0.46	0.45	0.47	0.49	0.51	0.46	0.48	0.90
SentiWN+WN-Affect	0.50	0.55	0.46	0.48	0.48	0.52	0.52	0.47	0.50	0.90
SentiWN + Micro-WNOp	0.50	0.56	0.49	0.45	0.49	0.50	0.52	0.47	0.50	0.90
All	0.53	0.57	0.51	0.48	0.52	0.52	0.55	0.50	0.52	0.91

5 Discussion

From Table 2, we can observe that the worst results for the word-based approach were obtained using WN-Affect. Note that the categories that are built from this resource contain few words and therefore the coverage of the method is affected (see Table 1 for statistics of the resources). For Micro-WNOp the method improves the coverage but fails in the detection of negative quotes (see low values of R- and F1-). The best results were obtained in the combinations that uses SentiWN; more negative opinions are correctly classified and better coverage is achieved. Combining SentiWN with other resources do not seem to improve the F1 scores, even though the coverage is slightly better. As WN-Affect and Micro-WNOp were built annotating a subset of WordNet senses and SentiWN includes all of these senses, it is likely that the four categories of SentiWN and those built from each combination are not significantly different (see Table 1).

Regarding the approach based on word sense disambiguation, we can observe in Table 3 that for the Micro-WNOp and WN-Affect resources the method obtains very low results, due to the low coverage of the annotated senses; from 115425 synsets in WordNet, only 1105 (0.96%) and 884 (0.77%) are annotated

on these resources, respectively. Also, the corpus has 1472 words, of which 1277 are non stop-words and are disambiguated with the senses of WordNet; Micro-WNOp only covers 57 (4.46%) and WN-Affect 18 (1.41%) of these ambiguous words. In spite of the low coverage, the method obtains acceptable precision values when these resources are used. For these reasons, when we use these resources individually, only a few words obtain a value of polarity. On the other hand, the use of SentiWN significantly improves both the F1 scores and the coverage.

Note also, that the combination of several resources obtains better precision, recall and F1 scores. Due to the fact that SentiWN was not manually annotated, some senses are misclassified (e.g., the sense FLU#1 (*an acute febrile highly contagious viral disease*) is annotated as Positive = 0.75, Negative = 0.0 and Objective = 0.25, despite having a lot of negative words in its gloss). These mistakes affect the polarity classification. We suppose that combining WN-Affect and Micro-WNOp with SentiWN reduces this negative influence, and consequently the precision, recall and F1 values are improved. The low coverage of the Micro-WNOp and WN-Affect resources do not allow higher increases in the classification quality.

Finally, Figure 1 shows the comparison of both methods (with and without WSD) for each resource combination with respect to overall F1 measure. As can be seen, the results of the method based on word sense disambiguation are better than those of the bag-of-words approach, except for Micro-WNOp where the WSD-based method is severely affected by the low coverage of this resource. The best F1 score of the WSD-based method is 0.52 and that of the method without WSD is 0.47. Note also that the WSD-based method not only obtains better overall F1, but also a higher coverage (see Tables 2 and 3). This confirms that word sense disambiguation is useful for determining the polarity of a word.

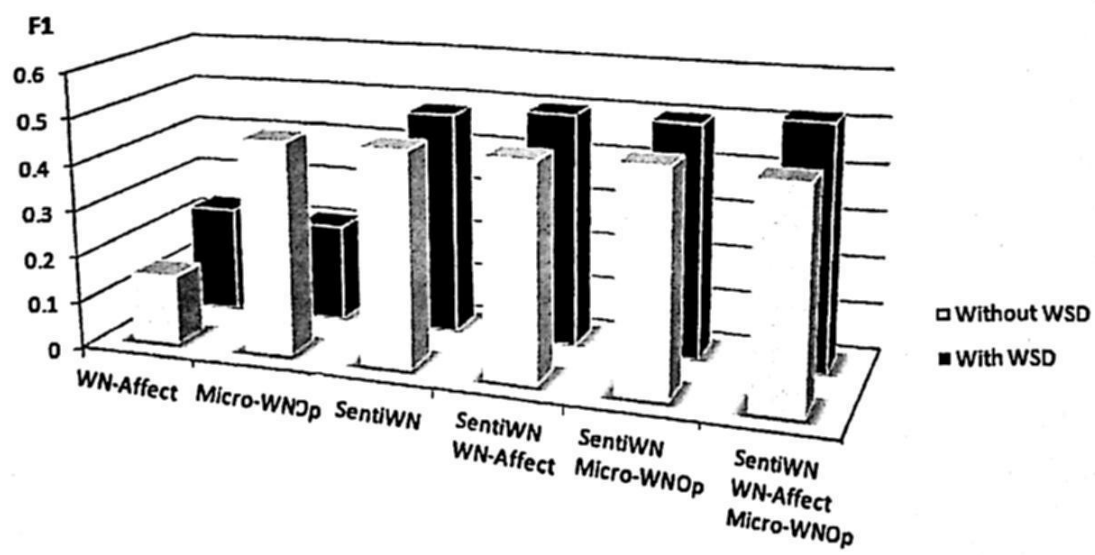

Fig. 1. Overall F1 scores for both methods.

Unlike the WSD-based method, the addition of WN-Affect and Micro-WNOp with respect to use only SentiWN does not contribute to improve the results of the method without WSD. In the first method, this is due to the higher quality in the sense polarity annotation, while in the second one to the absence of differences among the built categories.

From the results of Tables 1 and 2, we can also notice that Micro-WNOp and WN-Affect achieve better overall precision but lower overall recall than SentiWN in both methods. It can be expected due to the manual annotation and the low coverage of these resources, respectively.

Another interesting observation is that the bag-of-words approach leads to better performance when classifying positive quotes, whereas the WSD-based method achieves a good balance between the classification of positive quotes and the negative ones instead. This can be seen in the precision, recall and F1 values for each class.

6 Conclusions and Future Work

In this paper, a comparison between two unsupervised methods for determining the polarity of opinions has been presented. One of them performs the analysis at a word level, whereas the other at a sense level. Studies of the behaviour of both methods were presented over the same corpus and using several public resources for opinion mining.

The use of word sense disambiguation in the polarity classification has pros and cons. The advantages are given in the superiority of the results (with respect to precision, recall and F1) obtained by taking into account the context of the words appearing in the opinion. Also, the polarity detection has the same behavior in both classes, that is, the performance is balanced when positive and negative quotes are classified. Despite that some of the resources have a low coverage, this method obtains a better coverage than the bag-of-words approach.

However, as word sense disambiguation constitutes an intermediate task in the polarity classification, the disambiguation errors could affect the classification quality. This provides further motivation to study in depth this problem, due to the lack of a corpus manually annotated with senses and polarity. Also, the disambiguation algorithm depends on the used knowledge resources and, as we can saw in the experiments, there are no resources that have both high coverage and a good quality in the annotation of sense polarities.

Future work includes the study of alternative methods to extract and classify opinions, working at a syntactic level, or using local contexts, semantic representations of concepts and the modelling of discourse structures. Our idea is to study in a broader context the impact of word sense disambiguation on the performance of opinion mining systems, be it in small texts (such as the ones we have studied in this paper) or larger contexts (on-line discussion forums, blogs or newspaper articles). Another interesting application would be the determination of figurative senses, which are used to express opinions in a more sophisticated

manner. To this aim, the application of word sense disambiguation is an essential step.

References

1. Esuli, A., Sebastiani, F.: Sentiwordnet: A publicly available lexical resource for opinion mining. In: Fifth international conference on Language Resources and Evaluation (LREC 2006). (2006) 417–422
2. Hatzivassiloglou, V., McKeown, K.R.: Predicting the semantic orientation of adjectives. In: 35th Annual Meeting of the Association for Computational Linguistics, Madrid, Spain, The Association for Computational Linguistics (1997) 174–181
3. Turney, P., Littman, M.: Measuring praise and criticism: Inference of semantic orientation from association. ACM Transactions on Information Systems **21** (2003) 315–346
4. Kim, S., Hovy, E.: Determining the sentiment of opinions. In: 20th International Conference on Computational Linguistics (ACL 2004), Morristown, NJ, USA, The Association for Computational Linguistics (2004) 1367–1373
5. Takamura, H., Inui, T., Okumura, M.: Extracting emotional polarity of words using spin model. In: 43rd Annual Meeting of the Association for Computational Linguistics, Ann Arbor, US, The Association for Computational Linguistics (2005) 133–140
6. Esuli, A., Sebastiani, F.: Pageranking wordnet synsets: An application to opinion mining. In: 45th Annual Meeting of the Association for Computational Linguistics, Prague, Czech Republic, The Association for Computer Linguistics (2007) 424–431
7. Wiebe, J., Mihalcea, R.: Word sense and subjectivity. In: 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, Sydney, Australia, The Association for Computational Linguistics (2006) 1065–1072
8. Wiebe, J., Wilson, T., Cardie, C.: Annotating expressions of opinions and emotions in language. Language Resources and Evaluation **1** (2005) 165–210
9. Akkaya, C., Wiebe, J., Mihalcea, R.: Subjectivity word sense disambiguation. In: Conference on Empirical Methods in Natural Language Processing, Singapore, The Association for Computational Linguistics (2009) 190–199
10. Rentoumi, V., Giannakopoulos, G.: Sentiment analysis of figurative language using a word sense disambiguation approach. In: International Conference on Recent Advances in Natural Language Processing (RANLP 2009), The Association for Computational Linguistics (2009)
11. Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: Conference on New Methods in Language Processing. (1994) 44–49
12. Balahur, A., Steinberger, R., v. d. Goot, E., Pouliquen, B., Kabadjov, M.: Opinion mining on newspaper quotations. Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference **3** (2009) 523–526
13. Fellbaum, C.: WordNet: an electronic lexical database. MIT Press (1998)
14. Strapparava, C., Valitutti, A.: Wordnet-affect: an affective extension of wordnet. In: 4th International Conference on Language Resources and Evaluation, LREC 2004. (2004) 1083–1086
15. Cerini, S., Compagnoni, V., Demontis, A., Formentelli, M., Gandini, G.: Micro-WNOp: A gold standard for the evaluation of automatically compiled lexical resources for opinion mining. In: Language resources and linguistic theory: Typology,

second language acquisition, English linguistics. Franco Angeli Editore, Milano, IT (2007)
16. Martín-Wanton, T., Pons-Porrata, A., Montoyo-Guijarro, A., Balahur, A.: Opinion polarity detection: Using word sense disambiguation to determine the polarity of opinions. In: 2nd International Conference on Agents and Artificial Intelligence, Volume 1 - Artificial Intelligence, Valencia, Spain, INSTICC Press (2010) 483–486
17. Agirre, E., Edmonds, P.: Word Sense Disambiguation: Algorithms and Applications (Text, Speech and Language Technology). Volume 33. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2006)
18. Anaya-Sánchez, H., Pons-Porrata, A., Berlanga-Llavori, R.: Word sense disambiguation based on word sense clustering. In Coelho, J.S.H., Oliveira, S., eds.: Lecture Notes in Artificial Intelligence. Volume 4140., Springer (2006) 472–481
19. Gil-García, R., Badía-Contelles, J.M., Pons-Porrata, A.: Extended star clustering algorithm. In Sanfeliu, A., Ruiz-Shulcloper, J., eds.: Lecture Notes in Computer Sciences. Volume 2905., 8th Iberoamerican Congress on Pattern Recognition (CIARP), Springer-Verlag (2003) 480–487

Improving Emotional Intensity Classification using Word Sense Disambiguation

Jorge Carrillo de Albornoz[1], Laura Plaza[2], Pablo Gervás[3]
Computer Science Faculty, Universidad Complutense de Madrid,
C/ Prof. José García Santesmases, s/n. 28040, Madrid, Spain
[1]jcalbornoz@fdi.ucm.es, [2]lplazam@fdi.ucm.es, [3]pgervas@sip.ucm.es

Abstract. During the last years, sentiment analysis has become a very popular task in Natural Language Processing. Affective analysis of text is usually presented as the problem of automatically identifying a representative emotional category or scoring the text within a set of emotional dimensions. However, most existing approaches determine these categories and dimensions by matching the terms in the text with those presented in an affective lexicon, without taking into account the context in which these terms are immersed. This paper presents a method for the automatic tagging of sentences with an emotional intensity value, which makes use of the WordNet Affect lexicon and a word sense disambiguation algorithm to assign emotions to concepts rather than terms. An extensive evaluation is performed using the metrics and guidelines proposed in the SemEval 2007 Affective Text Task. Results are discussed and compared with those obtained by similar systems in the same task.

Keywords: Sentiment Analysis, Word Sense Disambiguation, Emotional Intensity, Machine Learning Techniques

1 Introduction

Sentiment analysis is becoming increasingly important in recent years. This discipline comprises very distinct research areas, such as text analysis, speech and facial expression, which have motivated a great number of systems, each one with specific properties and frequently using *ad hoc* and rarely available resources.

Focusing on text applications, emotional analysis usually relies on the identification of emotional keywords in the text [1]. These emotional keywords are either compared with those existing in an affective lexicon or used as the keys of a set of rules. Furthermore, most approaches work at the lexical level, using words or stems as emotional keywords, instead of the appropriate concepts according to their contexts. The few approaches that work at a conceptual level rarely make use of word sense disambiguation techniques in order to obtain the correct senses of the concepts. Instead, they simply obtain the first meaning or all possible meanings [2].

On the other hand, most sentiment analysis systems are focused on the identification of emotional categories or dimensions in the sentences [3, 4]. This information may be insufficient or even irrelevant in some applications. For instance,

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 131-142

Received 26/11/09
Accepted 16/01/10
Final version 08/03/10

the emotional intensity as well as its polarity (positive vs. negative) can be of interest in public opinion analysis systems [5].

In this paper, an emotional intensity classifier capable of determining the intensity and polarity of the sentences in a text is presented. This system has been conceived to be used in an automatic camera management system for virtual environments based on the emotional analysis of the film script. In this context, the emotional intensity of the scene is an important parameter if the adequate camera position and movements need to be selected from the dialogs and actions described in the film script. The system is based on the identification of concepts in the sentences rather than terms, using a word sense disambiguation tool to obtain the correct senses for these concepts. The WordNet Affect lexicon is used to identify those concepts which are a priori candidates to denote an emotion or feeling.

The paper is organized as follows. Section 2 exposes the background and related work on sentiment analysis. Section 3 presents the method proposed for the tagging of sentences with emotional intensities. Section 4 introduces the evaluation methodology as well as the results obtained and the comparison with other similar systems. In section 5, the experimental results are discussed. Finally, section 6 provides concluding remarks and identifies pending problems and future work.

2 Background

This section presents the background of this study, as well as some recent works on sentiment analysis.

2.1 Emotion

Nowadays the most outstanding psychological theories on sentiment analysis are the emotional dimensions theory and the emotional categories theory. The first theory proposes the interpretation of human emotions throughout a set of emotional dimensions. This theory is based on James Russel studies [6], where the human emotional space is presented as a circular bipolar structure that represents the *pleasure* and the *activation*. The emotional dimensions were employed in the development of the ANEW list (*Affective Norms for English Words*) according to the SAM standard (*Self Assessment Manikin*) [7], where each English word is assigned a value, between 1 and 9, in each of the three dimensions proposed: *pleasure, arousal* and *dominance*. This work was supported by the idea that humans understand emotions as opposite poles.

On the other hand, the emotional categories theory exposes the emotions as entities, primitive units with boundaries that can be countable [8]. This idea can be traced to Rene Descartes [9], who proposes a set of primitive emotions from which the other emotions can be derived. In order to represent these categories, this theory makes use of the everyday language (i.e. *joy* and *anger* are emotional categories). The main handicap of this theory is the disagreement on the most adequate set of emotion categories. While some works argue that a small set of categories is the most suitable selection [10], others studies expose that a bigger hierarchical set of categories is

necessary to enclose the rich human feeling [11]. This debate is also encouraged by the specific purpose of the studies. For instance, Ekman [12] argues that only six basic emotions (*anger, disgust, fear, joy, sadness* and *surprise*) are needed to analyze facial expressions, while Ortony et *al.* [13] present a set of 22 emotions in their OCC standard to emotion synthesis in speech.

2.2 Corpora and Affective Lexical Dictionaries

According to the different emotional theories, a wide range of resources has been developed, each of one supported by a psychological study and most of them specific for a given task. Focusing on the text sentiment analysis, any system that attempts to identify the affective meaning of a text will need, at least, two types of NLP resources: an annotated corpus to trail and test the system performance and an affective lexical dictionary that attaches affective meanings to words.

It is difficult to find affective corpora publicly available for researchers. Besides, most of them are very specific to the task and domain to which they have been designed; so that it is either impossible or surprisingly difficult to use them in a different task. An example of emotional corpus is Emotag [14]. Emotag consists of a set of sentences extracted from eight popular tales marked up by human evaluators with an emotional category and values for three emotional dimensions (*evaluation*, *activation* and *power*). A radically different corpus is the one proposed for the SemEval 2007 Affective Text task [15], where a set of 1250 sentences from news headings are manually tagged with six basic emotions (*anger, disgust, fear, joy, sadness* and *surprise*). Each emotion is scored between 0 and 100, where 0 indicates that the emotion is not present in the headline, and 100 indicates that the maximum amount of emotion is found in the headline. This corpus also includes a *valence* value in the interval [-100, 100] for each sentence, which expresses the negative (-100), neutral (0) or positive (100) intensity of the sentence.

Similarly to the corpora, the affective lexical dictionaries strongly depend on the underlying psychological theory. In relation to the emotional dimensions theory, the most popular lexicons are the ANEW word list (*Affective Norms for English Words*) [7] and the DAL dictionary (*Whissell's Dictionary of Affect Language*) [16]. The first one consists of a list of words scored within three emotional dimensions: *pleasure*, *arousal* and *dominance*, according to the SAM standard; while the second one contains 8742 words rated by people for their *activation*, *evaluation* and *imagery*. In relation to the emotional categories theory, the LIWC Dictionary (*Linguistic Inquiry and Word Count Dictionary*) [17] provides a set of 2290 words and stems, classified in one or more categories, such as *sadness*, *negative emotion* or *overall affect*.

The main limitation of these lexicons is the use of words or stems, instead of concepts, as the primitive units, without recognizing the context in which the words are used. On the contrary, the WordNet Affect database [18] provides a list of 911 WordNet synsets labeled with a hierarchical set of emotional categories. Most of the synsets labeled are representative of the meaning of the emotional categories (nouns and adjectives), while others are suitable to denote affective meanings.

2.3 Sentiment Analyzers

As already mentioned, the most accepted approach to sentiment analysis is the identification of a set of emotional keywords in the text. However, a great variety of methods have been proposed to achieve this purpose. Francisco and Gervás [14] present a system which is consistent with both theories: the emotional categories and the emotional dimensions, based on the averaged frequencies of the words found in a corpus of tales. Subasic and Huettner [19] propose a fuzzy approximation that uses an affective lexicon containing words manually annotated with two properties: *centrality* and *intensity*. These properties are, respectively, the membership degree of an emotional category and the strength of the affective meaning. A similar approach is presented in [20] where the emotional categories are represented as fuzzy hipercubes with a range of intensity between 0 and 1.

More sophisticated are the methods that aim to improve the emotional information extracted by means of semantic rules and syntactical analysis. Zhe and Boucouvalas [21] present an emotion extraction engine that uses a parser to identify auxiliary verbs, negations, subject of the sentence, etc. Wu et *al.* [3] expose a system where the annotation process is guided by emotional generation rules manually deduced from psychology. In the same line, Mostafa Al Masum et *al.* [22] present the system ASNA (*Affective Sensitive News Agent*) that uses the OCC model with a set of rules and natural language processing techniques to automatically classify news. Nicolov et *al.* [23] analyzed the effect of coreference resolution in sentiment analysis.

A third common approximation to the problem is the use of Machine Learning techniques. Devillers et *al.* [24] study the applicability of different machine learning algorithms to identify relevant emotional states in real life spoken interactions. In contrast, Seol et *al.* [25] present a hybrid system that uses emotional keywords if these are present in the text or a set of domain knowledge-based artificial neural networks if no emotional keywords are found. The neural networks are initialized with a set of rules that determine possible emotional categories states.

3 The Emotion Classifier

In this section, the method for automatically labeling sentences with an emotional intensity is presented. The problem is faced as a text classification task. The classifier aims to identify the emotional intensity of each sentence in a text, as well as if this intensity denotes a positive or negative emotional meaning. The method accomplishes the task throughout four steps. Each step is explained in detail in the following subsections, along with a working example that illustrates the algorithm. Besides, in order to clarify how the system works and what resources are used, its architecture is shown in Fig. 1.

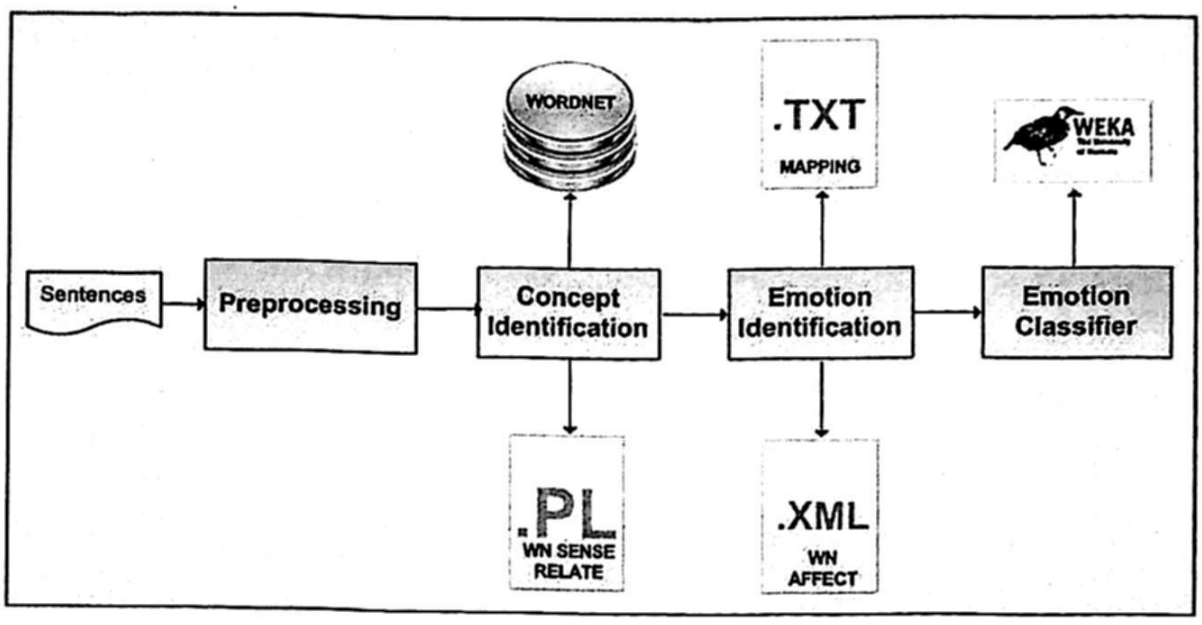

Fig. 1. Architecture of the emotional intensity tagger

3.1 Preprocessing

As most NLP systems, a preliminary preprocessing of the input text is needed. This includes splitting the text into sentences and tagging the words with their part of speech (POS). To this purpose, the *Tokenizer, Part of Speech tagger* and *Sentence splitter* modules in GATE [26] have been used. Generic and high frequency terms are removed using a stop list. Besides, as only the *nouns*, *verbs*, *adjectives* and *adverbs* can present an emotional meaning, only terms from these grammatical categories are considered.

3.2 Concept Identification

Once the text has been split into sentences and the words have been labeled with their POS, the next step is the mapping of the terms in the sentences to their appropriated concepts in the *WordNet* lexical database [27].

In order to correctly translate these terms to WordNet concepts, a word sense disambiguation tool is needed. To this aim, the implementation of the *lesk* algorithm in the *WordNet Sense Relate* Perl package was used [28]. As a result, for each word its corresponding stem and sense in WordNet are obtained. This information is used to retrieve the appropriate synset that represents the concept in WordNet. Next, the hypernyms of each concept are also retrieved.

Fig. 2 shows the concept identification process for the sentence: *Foetal mechanism helps heart failure.* In this sentence, the term *foetal* has not been correctly disambiguated by WordNet Sense Relate and its synset could not be retrieved from WordNet.

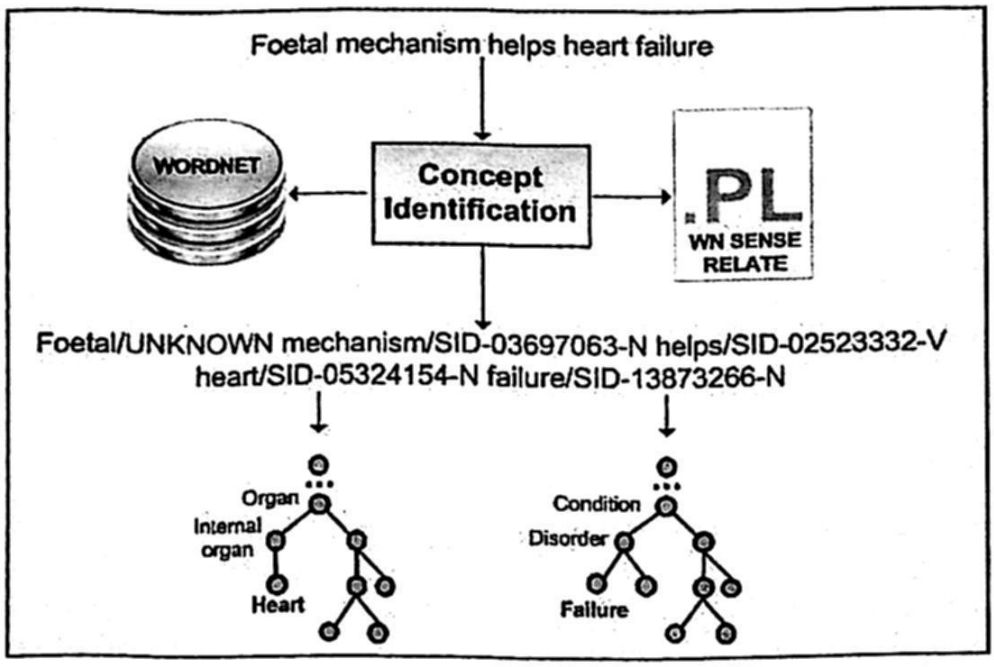

Fig.2. An example of the concept identification process

3.3 Emotion Identification

The goal of this third step is to match the previously identified WordNet concepts to their emotional categories in the WordNet Affect affective lexicon.

Focusing on the WordNet Affect emotion sub hierarchy, the first level distinguishes between *positive-emotions*, *negative-emotion*, *neutral-emotions* and *ambiguous-emotions,* while the second level encloses the emotional categories themselves. This level contains most of the basic emotions exposed in the emotional categories theories, such as *sadness, joy* and *surprise.* As the hierarchy used in WordNet Affect is considerably broader than those frequently used in sentiment analysis, the authors have identified that the second level is a good representation of human feeling and a good starting point for the attribute selection for the classifier and its evaluation. This subset contains 32 emotional categories.

Thus, once the synset of each word in the sentence has been identified, its emotional category is retrieved from WordNet Affect (if the concept appears in the lexicon). The same analysis is carried out over their hypernyms if no entry in WordNet Affect is found for the synset. To this aim, a previous mapping between synsets of WordNet 2.1 and WordNet 1.6 versions is needed, since the method and the affective lexicon works on different versions of WordNet. The use of the WordNet 2.1 version instead of the WordNet 1.6 is motivated by the fact that this version is the most updated for windows operative systems.

This process is illustrated in Fig. 3. It can be observed that only two concepts in the example sentence have been assigned an emotional meaning after this step. The emotional category for the concept *helps* is retrieved from its own synset, which is assigned the *liking* category. In contrast, the synset of the concept *failure* is not labeled in WordNet Affect, so the analysis of its hypernyms is carried out. As it first

level hypernyms (*disorder*) is labeled with the *general-dislike* category, this same category is assigned to *failure*.

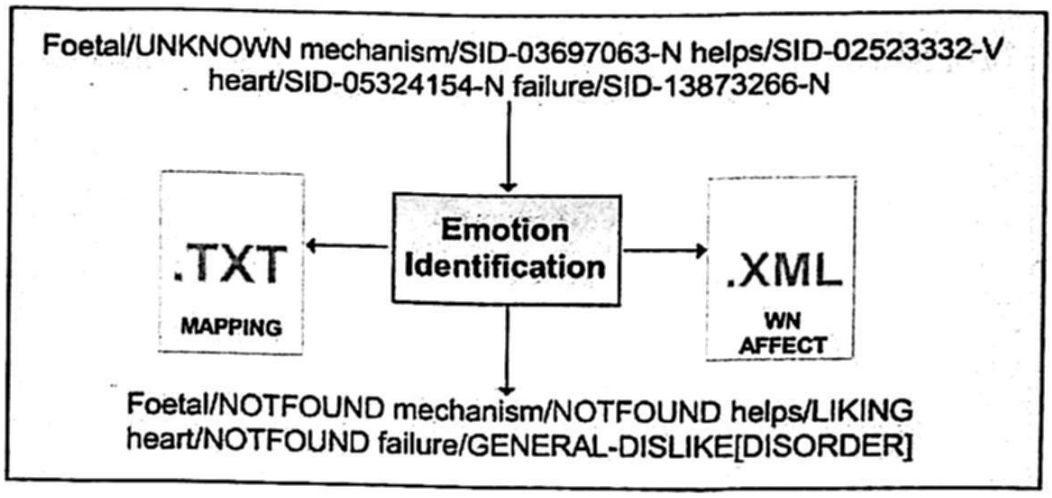

Fig. 3. An example of the emotion identification process

3.4 Emotional Intensity Classification

Up to this point, all the words in the sentence have been labeled with their emotional category (if any). Next, a vector of emotional occurrences (*VEO*) is created, which will be used as input for the *Random Forest* classifier implementation as provided by the *Weka* machine learning tool [29].

A VEO vector is an array of 32 positions, one representing each emotional category. To construct this vector, each concept is evaluated in order to determine its degree of affective meaning. If the concept has been assigned an emotional category, then the position in the VEO vector that represents that category is increases in 1. If no emotional category was retrieved for the concept, then the nearest labeled hypernym is used. As a hypernym is a generalization of the concept, a lower weight is assigned to the category position in the VEO vector, which depends on the depth of the hypernym in the hierarchy, as defined in (1). In order to avoid an excessive emotion generalization, only the first n levels of hypernyms are considered, where n has been empirically set to 3.

$$VEO\,[i] = VEO\,[i] + 1/(\text{Hyper. Depth}+1) \tag{1}$$

Fig. 4 shows the VEO vector for the example sentence. Since the emotional category *liking* is assigned to *helps*, the position of *liking* in the VEO vector is increased in one degree. On the other hand, the concept *failure* is labeled with the emotional category *general-dislike* through its first level hypernym, so its position in the VEO vector is increased in 0,5.

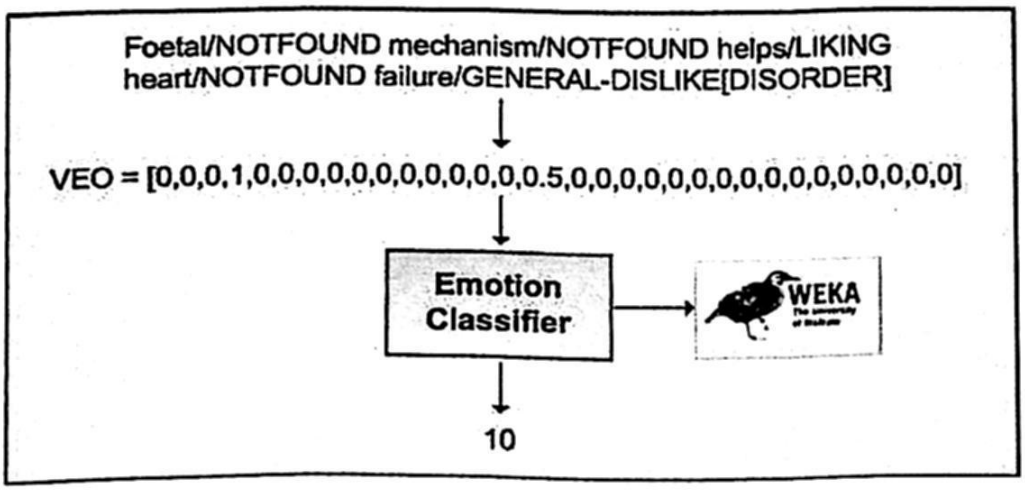

Fig. 4. An example of the emotional intensity classifier

Finally, the VEO vector is used as input for the classifier, and an intensity value between -100 and 100 is obtained as output.

4 Evaluation

In order to evaluate the performance of the method, a large–scale evaluation is accomplished following the guidelines observed in the SemEval 2007 Affective Text Task and using the corpus developed for this task.

The evaluation corpus consists of a training set of 250 sentences and a test set of 1000 sentences of news headlines. Each sentence has been manually labeled by experts with a score between -100 and 100, where -100 means a strongly negative emotional intensity, 100 means a strongly positive emotional intensity and 0 means neutral.

In this task, the intensity score assigned to each sentence has been mapped to three broader classes, -100 (between -100 and -50), 0 (between -50 and 50) and 100 (between 50 and 100). The results are evaluated using precision and recall metrics.

Due to the specific language used in the news domain most of the affective words in the headlines are not found in WordNet Affect. Thus, a set of emotional meanings has been manually added to the lexicon, such as *dead*, *kill* and *kidnap*. In particular, 188 synsets have been added: 140 nouns, 56 verbs and 22 adjectives.

To determine the best classification algorithm, all the experiments have been carried out over the different classification techniques implemented in Weka, although only the results of the best four algorithms are presented here.

The first group of experiments is directed to find out the best value for the number of hypernym levels as explained in section 3.4. For this parameter, we have considered all possible values from 0 to 5. As it can be observed in Table 1, using three levels of hypernyms produces the best results in 2 out of the 4 algorithms.

Table 1. Evaluation of the number of hypernym levels

Algorithm		Number of hypernyms level					
		0	*1*	*2*	*3*	*4*	*5*
Functional Trees	*Precision*	52,2	57,0	58,1	**60,4**	60,2	60,2
	Recall	62,2	62,2	62,8	**63,1**	62,9	62,8
Random Forest	*Precision*	57,5	57,0	57,4	**57,6**	57,4	57,3
	Recall	62,0	62,0	62,1	**62,5**	62,3	62,2
Naïve Bayes	*Precision*	56,3	**58,5**	58,3	58,3	58,3	58,3
	Recall	60,1	**59,8**	59,7	59,5	59,5	59,5
Multinomial Logistic	*Precision*	55,8	57,5	57,4	57,4	57,8	**58,1**
	Recall	62,0	62,6	62,9	63,1	63,1	**63,2**

The aim of the second group of experiments is to determine the best set of emotional categories for the classifier. Starting from the initial set of 32 categories, three algorithms for attribute selection implemented in Weka have been applied, which lead to three subsets of 3, 4 and 12 attributes respectively. Next, the four classifiers have been evaluated over these reduced sets of attributes. For these experiments, the number of hypernym levels was set to 3. Table 2 shows that two classifiers perform better with the subset of attributes selected by the *CFS Best First* algorithm.

Table 2. Evaluation of the set of attributes

Algorithm		Consistency Best First	CFS Best First	Consistency Genetic
Functional Trees	*Precision*	**57,7**	57,0	57,2
	Recall	**62,8**	62,2	62,3
Random Forest	*Precision*	63,1	**64.0**	63,5
	Recall	63,1	**63,5**	63,5
Naïve Bayes	*Precision*	55,8	**58,7**	55,8
	Recall	62,4	**61,3**	62,4
Multinomial Logistic	*Precision*	**57,7**	57,0	57,3
	Recall	**62,8**	62,2	62,5

A third group of experiments has been carried out in order to evaluate the effect of the distribution of the intensity within classes. To this aim, the emotional intensity of the sentences has been mapped to 3 balanced classes: -100 (from -100 to -35), 0 (from -35 to 35) and 100 (from 35 to 100), and to 5 balanced classes: -100 (-100 to -60), -50 (-60 to -20), 0 (-20 to 20), 50 (20 to 60) and 100 (60 to 100). For these experiments, the number of hypernym levels was set to 3, while the subset of attributes selected by the *CFS Best First* algorithm was used. Table 3 shows that the method performance decreases substantially when the number of classes is increased, while only a small reduction is reported when the 3 classes are equitably distributed with respect to the original distribution.

Therefore, the best configuration implies using the *Random Forest* algorithm along with the 4 attribute subset obtained by the *CFS Best First* algorithm and 3 hypernym levels.

Table 3. Evaluation of the intensity distribution in classes

Algorithm		3 Classes	5 Classes
Functional Trees	*Precision*	54,8	26,7
	Recall	51,0	35,5
Random Forest	*Precision*	**55,4**	36,6
	Recall	**52,1**	35,7
Naïve Bayes	*Precision*	52,1	**39,1**
	Recall	43,9	**32,9**
Multinomial Logistic	*Precision*	55,9	26,7
	Recall	50,7	35,6

Finally, Table 4 summarizes the best results of our method along with those of the systems participating in the SemEval 2007 Affective Text Task. Our method outperforms the other systems in precision, while CLaC-NB obtains the best recall.

Table 4. Comparison with SemEval 2007 systems

Systems	Precision	Recall
CLaC	61.42	9.20
UPAR7	57.54	8.78
SWAT	45.71	3.42
CLaC-NB	31.18	**66.38**
SICS	28.41	60.17
Our method	**64.00**	63.50

5 Discussion

As already mentioned, the system has obtained very promising results. When compared to other systems that take part in the SemEval task, our method obtains the best precision (64%), while providing the second highest recall (63.5%). Furthermore, both metrics are well balanced, which does not occur in the other systems.

The use of concepts instead of terms, along with the use of a word sense disambiguation algorithm allows the method to correctly map the words in the sentence to their emotional categories in the affective lexicon. Besides, the use of hypernyms has permitted to increase the number of concepts labeled with an emotion, which has significantly improved the evaluation results. This indicates that the method is strongly dependent on the number of concepts labeled in the lexicon.

Another interesting result is that the initial set of 32 emotional categories has proved to be too large. Some of these categories introduces noise and decreases the efficiency of the classification algorithms. According to the experiments, the best set consists of 4 emotional categories: *liking, negative-fear, sadness* and *general-dislike.* However, it is important to note that this subset will depend on the regarded domain.

A detailed examination of the precision and recall obtained by each intensity class has shown that the positive class encloses most of the classification errors. The reason seems to be that a good number of the positive sentences are expressed as the

negation of negative emotional concepts (i.e. *Jet flips in snowstorm, none dead*). A previous process of negation detection which adapts negated sentences to a positive form through antonym relations could minimize this problem.

6 Conclusions and Future Work

In this paper, an effective approach to automatically assign an emotional intensity and polarity to a sentence is presented. The method obtains both high precision and recall, which are also well balance. The evaluation results outperform those obtained by the systems participating in the SemEval 2007 Affective Text Task.

However, several problems have been identified. First, the experimentation has shown that the amount of index concepts is low, since most concepts in the corpus cannot be retrieved from WordNet Affect. This clearly has an influence on the precision of the method, which can be improved by enriching the lexicon with the corpus specific vocabulary.

A second handicap to the concept identification is the part of speech tagging errors reported by the GATE POS tagger, which has resulted in an good number of concepts that could not be correctly disambiguated and retrieved from WordNet. Future work will include a further evaluation using the statistical Stanford parser [30].

Finally, different techniques will be studied in order to detect negated concepts, as well as their scope, since they can invert the polarity of the sentences.

Acknowledgments. This research is funded by the Spanish Ministry of Science and Innovation (TIN2009-14659-C03-01). It is also partially funded by the Comunidad Autonoma de Madrid (CAM) and the European Social Fund (ESF) through the IV PRICIT program, and by the Spanish Ministry of Science and Innovation through the FPU program.

References

1. Strapparava, C., Mihalcea, R.: Learning to identify emotions in text. In: Proceedings of the 2008 ACM symposium on Applied computing, pp. 1556--1560. Ceara, Brazil (2008)
2. Chaumartin, F.R.: UPAR7: a knowledge-based system for headline sentiment tagging. In: Proceedings of the 4th International Workshop on Semantic Evaluations, pp. 422--425. Prague, Czech Republic (2007)
3. Wu, C.H., Chuang, Z.J., Lin, Y.C.: Emotion recognition from text using semantic labels and separable mixture models. Journal of ACM Transactions on Asian Language Information Processing 5, pp. 165--183 (2006)
4. Rubin, V.L., Stanton, J.M., Liddy, E.D.: Discerning Emotions in Texts. In: Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications. Stanford, C.A. (2004)
5. Mullen, T., Collier, NSentiment Analysis using Support Vector Machines with Diverse Information Sources. In: Proceedings of EMNLP, pp. 412--418. Barcelona, Spain (2004)
6. Russell, J. A.: A circumplex model of affect. Journal of Personality and Social Psychology 36, pp. 1161--1178 (1980)

7. Bradley, M.M., Lang, P.J.: Affective norms for English words (ANEW): Instruction manual and affective ratings. Technical Report C-1, The Center for Research in Psychophysiology, University of Florida (1999)
8. James, W.: What is an Emotion? Published in Mind 9, pp. 188--205 (1884)
9. Anscombe, E., Geach, P.T.: Descartes: Philosophical Writings. Ed. Nelson University Paperbacks (1972)
10. Plutchik, R.: A general Psycho Evolutionary Theory of Emotion. Emotion: Theory, Research, and Experience (1980)
11. Parrott, W.G.: Emotions in Social Psychology: Essential Readings. Psychology Press, Philadelphia (2001)
12. Ekman, P.: Are there basic emotions? Journal Psychol. Rev. 99, pp 550--553 (1992)
13. Ortony, A., Clore, G. L., Collins, A.: The Cognitive Structure of Emotions. Cambridge University Press (1988)
14. Francisco, V., Gervás, P.: Ontology-Supported Automated Mark Up of Affective Information in Texts. Special Issue of Language Forum on Computational Treatment of Language 34, pp. 26--36 (2008)
15. SemEval 2007 Affective Text Task Corpus, http://www.cse.unt.edu/~rada/affectivetext/
16. Whissell, C.M.: The dictionary of affect in language. In: Robert Plutchik and Henry Kellerman (Ed.), Emotion: Theory, Research, and Experience, pp. 113--131 (1989)
17. Pennebaker, J.W., Francis, M.E., Booth, R.J.: Linguistic Inquiry and Word Count (LIWC): LIWC2001. Erlbaum Publishers, Mahwah, NJ (2001)
18. Strapparava, C., Valitutti, A.: Wordnet-affect: an affective extension of WordNet. In: Proceedings of the 4th International Conference on Language Resources and Evaluation, pp. 108--1086. Lisbon, Portugal (2004)
19. Subasic, P., Huettner, A.: Affect Analysis of Text Using Fuzzy Semantic Typing. Journal of IEEE Transactions on Fuzzy Systems 9, pp 483--496 (2001)
20. Esau, N., Kleinjohann, L., Kleinjohann, B.: An Adaptable Fuzzy Emotion Model for Emotion Recognition. In: Proceeding of EUSFLAT Conference, pp. 73--78 (2005)
21. Zhe, X., Boucouvalas, A.: Text-to-Emotion Engine for Real Time Internet Communication. In: Proceedings of International Symposium on CSNDSP, pp. 164--168 (2002)
22. Mostafa Al Masum, S., Islam, M. T., Ishizuka, M.: ASNA: An Intelligent Agent for Retrieving and Classifying News on the Basis of Emotion-Affinity. In: Proceedings of the CIMCA-IAWTIC'06 (2006)
23. Nicolov, N., Salvetti, F., Ivanova, S.: Sentiment Analysis: Does Coreference Matter? In: of the Symposium on Affective Language in Human and Machine, pp. 37--40. (2008)
24. Devillers, L., Vidrascu, L., Lamel, L.: Challenges in real-life emotion annotation and machine learning based detection. Journal of Neural Netw. 18, pp 407--422 (2005)
25. Seol, Y.S., Kim, D.J., Kim, H.W.: Emotion Recognition from Text Using Knowledge-based ANN. In: Proceedings of 23rd International Technical Conference on Circuits/Systems, Computers and Communications, pp. 1569--1572 (2008)
26. General Architecture for Text Engineering, http://gate.ac.uk/
27. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to WordNet: An On-Line Lexical Database. International Journal of Lexicography 3 (4), pp. 235--244 (1990)
28. Patwardhan, S., Banerjee, S., Pedersen, T.: SenseRelate::TargetWord - A Generalized Framework for Word Sense Disambiguation. In: Proceedings of the Twentieth National Conference on Artificial Intelligence (Intelligent Systems Demonstrations), pp. 1692–1693. Pittsburgh, PA (2005)
29. Weka Machine Learning Tool, http://www.cs.waikato.ac.nz/ml/weka/
30. Statistical Stanford Parser, http://nlp.stanford.edu/software/lex-parser.shtml

Sentence Level News Emotion Analysis in Fuzzy Multi-label Classification Framework

Plaban Kr. Bhowmick, Anupam Basu, Pabitra Mitra and Abhisek Prasad

Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Kharagpur, India-721302
plaban@gmail.com, anupambas@gmail.com, pabitra@gmail.com,
abhisek.hi@gmail.com

Abstract. Multiple emotions are evoked with different intensities in readers' minds in response to text stimuli. In this work, we perform reader perspective emotion analysis in sentence level considering each sentence to be associated with the emotion classes with fuzzy belongingness. As news articles present emotionally charged stories and facts, a corpus of 1305 news sentences are considered in this study. Experiments have been performed in Fuzzy k Nearest Neighbor (FkNN) classification framework with four different feature groups. Word feature based classification model is considered as baseline. In addition to that, we have proposed three features namely, polarity, semantic frame and emotion eliciting context (EEC) based features. Different measures applicable to multi-label classification problem have been used to evaluate the system performance. Comparisons between different feature groups revealed that EEC based feature is the most suitable one in reader perspective emotion classification task.

1 Introduction

With the recent thrust in Human-Computer Interaction (HCI) and Human Centered Computing (HCC), researchers are concerned about modeling human behavior in order to provide truly intelligent interfaces. Emotion is one of the distinguishing features of human character and plays an important role in shaping human behavior. Current efforts in HCI area are exploring the possibilities of developing emotionally intelligent interfaces. *Emotional intelligence* refers to one's ability to understand and manage the emotion of one's self or of others. Apart from other modes like speech and facial expression, language is one of the most common modes for expressing emotion whether it is day-to-day speech communications (spoken language) or published communications (written language). Recent works in natural language processing area look into different behavioral aspects of human like personality trait, sentiment and emotion. Emotion can be studied from two perspectives.

- From the writer/speaker perspective, where we need to understand the emotion that the writer/speaker intended to communicate and

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 143-154

Received 23/11/09
Accepted 16/01/10
Final version 11/03/10

- from the reader's perspective, where we try to identify the emotion that is triggered in a reader in response to a language stimulus.

In this work, we intend to perform sentence level emotion classification from reader's perspective. The issues that are needed to be addressed in this kind of task are as follows:

- Fuzzy and multi-label characteristics: In response to a text stimuli, a blend of emotions may be evoked in reader's mind with different degrees of intensity. Thus, from a classification point of view, a text segment may have multiple memberships in different emotion categories.
- Identification of suitable features: As reader perspective emotion analysis is in its infancy, the exploration on the identification of suitable features has to be performed.
- Feature sparseness: While emotion elicitation from a discourse or paragraph may provide larger number of cues as features, the number of features available from a single sentence is less and hence the feature space becomes sparse.

The earlier works towards reader perspective emotion classification have used word feature [1, 2] and word co-occurrence statistics [3]. In this work, we have introduced three new features, namely the polarity feature, semantic frame feature and emotion elicitation context (EEC) feature towards the same objective. As fuzziness is involved in subjective entity like emotion, Fuzzy k Nearest Neighbor (FkNN) framework has been used for developing emotion classification models with proposed features.

2 Related Works

As stated earlier, emotion analysis can be performed in two different perspectives. There are a number of efforts towards writer perspective emotion analysis [4–8]. As we focus on performing reader perspective emotion analysis, we provide an overview of the works addressing the related task.

Affective text analysis was the task set in *SemEval-2007 Task 14* [9]. A corpus of news headlines extracted from Google news and CNN was provided. Two types of tasks were to classify headlines into positive/negative emotion category as well as distinct emotion categories like anger, disgust, fear, happiness, sadness and surprise.

The system UA-ZBSA [3] gathers statistics from three different search engines (MyWay, AllWeb and Yahoo) to attach emotion lables to the news headlines. The work computes the PMI score of each content word of a headline with respect to each emotion by querying the search engines with the headline and the emotion. The accuracy, precision and recall of the system is reported to be 85.72%, 17.83% and 11.27%.

UPAR7 [10] is a linguistic rule-based approach towards emotion classification. The system performs emotion analysis on news headline data provided in

SemEval-2007 Task 14. In the preprocessing step, the common words are decapitalized with the help of parts of speech tagger and Wordnet. Each word first is rated with respect to emotion classes. The main theme word is detected by parsing a headline and it is given a higher weight than the other words in the headline. The emotion score boosting to the nouns are performed based on their belongingness to some general categories in Wordnet. The word scoring also considers some other factors like human will, negation and modals, high-tech names, celebrities etc. The average accuracy, precision and recall of the system is 89.43%, 27.56% and 5.69%.

The system SWAT [11] adopts a supervised approach towards emotion classification in news headlines. A word-emotion map constructed by querying the Roget's New Millennium Thesaurus is used to score each word in the headline and the average score of the headline words are taken into account while labeling it with a particular emotion. The reported classification accuracy, precision and recall are 88.58%, 19.46% and 8.62%.

The work by Lin and Chen [1, 2] provides the method for ranking reader's emotions in Chinese news articles. Eight emotional classes are considered in this work. Chinese character bigram, Chinese words, news metadata, affix similarity and word emotion have been used as features. The best reported system accuracy is 76.88%.

3 Emotion Data

The emotion text data collected by us consists of 1305 sentences extracted from *Times of India* news paper archive[1]. The sentences were collected from headlines as well as from the bodies of articles belonging to political, social, sports and entertainment domain. The annotation scheme considers the following points:

- *Choice of emotion classes*: The annotation scheme considers four basic emotions, namely, *Disgust, Fear, Happiness, Sadness.*
- *Fuzzy and Multi-label annotation*: A sentence may trigger multiple emotions simultaneously. So, one annotator may classify a sentence to more than one emotion category. Fuzzy annotation is considered in this work, i.e., for a sentence, the annotators provide a value from the range [0,1] against each emotion category.

The distribution of sentences across emotion categories is as follows: Disgust = 307, Fear = 371, Happiness = 282 and Sadness = 735.

4 Features for Emotion Classification

Following features were considered in the experiments on emotion analysis on the data set described above.

[1] http://timesofindia.indiatimes.com/archive.cms

4.1 Word Feature

Words are sometimes indicative of the emotion class of a text segment. For example, the word 'bomb' may be highly co-associated with fear emotion. Thus, words present in the sentences may be considered to be potential features. Now, if we consider all the words in a text corpus, only a subset of these will be present in a particular sentence. The presence of these words is used to form a binary feature vector. Before creating the word feature vectors, following preprocessing steps are adopted.

- Stop words are removed.
- Named Entities may introduce noise in emotion classification. So, named entities are removed using the Stanford named entity recognizer[2].
- The remaining content words are stemmed using Porter's stemmer algorithm.

4.2 Polarity based Feature

Polarity of the subject, object and verb of a sentence may be good indicators of whether the sentence evokes positive or negative emotions. For example, let us consider the following sentence.

```
Relief work improves the poor conditions of flood affected people.
```

Here, the subject, *Relief work*, is of positive polarity; the verb, *improves*, is of positive polarity; and the object phrase, *poor conditions of flood affected people*, is of negative polarity. Intuitively, a positive subject performs a positive action on a negative object and this pattern evokes a positive emotion.

The polarity values of each word in the corpus are tagged manually. Existing resources like SentiWordnet may have been employed in word level polarity tagging. However, as this resource is developed using machine learning techniques, the error introduced in the polarity learning may affect the performance of emotion classification. The polarity of a word may have values like POSITIVE (P), NEGATIVE (Ne) or NEUTRAL (N).

The problem of finding polarity of verb and its corresponding subject and object in a sentence can be broken down into following sub-problems:

- Finding out the main verb and head words of the corresponding subject and object phrase
- Finding the modifier words for verb, subject and object head words
- Finding polarities of subject, object and verb phrases

The Stanford Parser[3] is used to parse the sentences and the dependency relations (nsubj, dobj, etc.) obtained as parser output are used to extract the subject, verb and object phrases. A dependency relation from the output of the parser is of the following form.

[2] http://nlp.stanford.edu/software/CRF-NER.shtml
[3] http://nlp.stanford.edu/software/lex-parser.shtml

```
relation(arg1, arg2)
```

The main verb, subject and object head words in a sentence is detected using the dependency relations obtained from the parser output. Some of these relations are given in Table 1.

Table 1. Example dependency relations for identification of verb, subject and object head words.

Relation	Argument	Example Sentence	Example Relation
Example dependency relations for identification of verb			
Agent: *agent*	arg1	The man has been killed by the police	agent(*killed*, police)
Passive auxiliary: *auxpass*	arg1	The president was killed	auxpass(*killed*, was)
Clausal subject: *csubj*	arg1	What he has done made me proud	csubj(*made*, done)
Direct object: *dobj*	arg1	He gave me a book	dobj(*gave*, book)
Nominal subject: *nsubj*	arg1	Millitants destroyed the bridge	nsubj(*destroyed*, millitants)
Passive nominal subject: *nsubjpass*	arg1	The bridge was destroyed by millitants	nsubjpass(*destroyed*, bridge)
Example dependency relations for identification of subject head word			
Agent: *agent*	arg2	The man has been killed by the police	agent(killed, *police*)
Clausal subject: *csubj*	arg2	What he has done made me proud	csubj(made, *done*)
Clausal passive subject: *csubjpass*	arg2	That he will excel was predicted	csubjpass(predicted, *excel*)
Nominal subject: *nsubj*	arg2	Millitants destroyed the bridge	nsubj(destroyed, *millitants*)
Controlling subject: *xsubj*	arg2	Tom likes to eat fish	xsubj(eat, *Tom*)
Example dependency relations for identification of object head word			
Direct object: *dobj*	arg2	He gave me a book	dobj(gave, *book*)

The second step is to find out the modifiers of the subject and object and verb head words. The example relations that were used for extracting the modifiers are given in Table 2. The polarity assignment to a phrase is performed with

Table 2. Example dependency relations for identification of modifier words

Relation	Argument	Example Phrase	Example Relation
Adverbial modifier: *advmod*	arg2	Genetically modified food	advmod(modified, *genetically*)
Adjectival modifier: *amod*	arg2	Genetically modified food	amod(food, *modified*)
Negation modifier: *neg*	arg2	He was not killed	negkilled, *love*

two different sets of phrase polarity assignment rules one for verb phrase (see Table 3)and another for subject and object phrase (see Table 4).

4.3 Semantic Frame Feature

Every word in lexicon refers to some ground truth conceptual meaning that helps in clustering the words based on their conceptual similarity. In frame semantics [12], a word evokes a frame of semantic knowledge relating to the specific concept it refers to.

Table 3. Example rules for verb polarity assignment (P = Positive, Ne = Negative, N = Neutral, NULL = absent, X = independent of relation)

Rule#	Head	Modifier	Relation	Phrase	Example
V1	Ne	Ne	advmod	Ne	[brutally]/Ne [killed]/Ne ⟶ [brutally killed]/Ne
V2	P	P	advmod	P	[heartily]/P [welcomed]/P ⟶ [heartily welcomed]/P
V3	Ne	P	advmod	Ne	[artistically]/P [murdered]/Ne ⟶ [artistically murdered]/Ne
V4	P	Ne	advmod	Ne	[ghastly]/Ne [welcomed]/P ⟶ [ghastly welcomed]/Ne
V5	N	P|Ne	advmod	Modifier polarity	[beautifully]/P [taken]/N ⟶ [beautifully taken]/P

Table 4. Example rules for subject and object polarity assignment

Rule#	Head	Modifier	Phrase	Example
N1	P	P	P	[great]/P [win]/P ⟶ [great win]/P
N2	Ne	P|Ne|N	Ne	[airplane]/N [hijack]/Ne ⟶ [airplane hijack]/Ne
N3	N	P|Ne	Modifier polarity	[excellent]/P [performance]/N ⟶ [excellent performance]/P
N4	N	N	N	[Minor]/N [girl]/N ⟶ [Minor girl]/N
N5	P|Ne	NULL	Head polarity	[bomb]/Ne ⟶ [bomb]/Ne

The Berkeley FrameNet project[4] is a well-known resource of frame-semantic lexicon for English. Apart from storing the predicate-argument structure, the frames group the lexical units. For example, the frame **Apply_heat** is evoked by the lexical units such as *bake, blanch, boil, simmer, steam,* etc. So, assignment of appropriate frames to the words may be used as a generalization technique.

The semantic frame feature extraction was performed by considering the semantic parse of each sentence through SHALMANESER[5]. In the example sentence given below, the words 'arrest', 'man', 'abducting', 'assaulting' and 'girl' are assigned with *Arrest, People, Kidnapping, Rape* and *People* frames. These frames are considered as semantic frame features.

```
Villivakkam police arrested a 26-year old married man for
abducting and sexually assaulting a 16-year-old girl
```

4.4 Emotion Elicitation Context (EEC) Feature

Emotion is evoked in reader based on the situation described in the text. The surface level features like words are not adequate to encode these situations. In order to represent these situations, we need to capture the context in which they occur. In order to capture these contexts, we develop a knowledge base of emotion eliciting contexts.

Representing EEC Knowledge An EEC involves an action and entities related to the action. One context is described by a semantic graph that contains a special node called the *pivot* representing the action part of EEC. The *pivot* node

[4] http://framenet.icsi.berkeley.edu/

[5] http://www.coli.uni-saarland.de/projects/salsa/shal/

is reference to a semantic frame like *Cause_harm* of Framenet. The entity nodes in the semantic graph are related to the *pivot* node with semantic relations. The entity nodes are reference to the semantic groups (SG). An SG is a collection of similar semantic frames or concepts or both. The g^{th} SG is represented as follows.

$$SG_g : SF_1, SF_2, \ldots, SF_s; C_1, C_2, \ldots, C_c \quad (1)$$

where SG_g contains s number of semantic frames and c number of concepts. The s^{th} semantic frame SF_s is reference to a frame in FrameNet. There are some terms which cannot be mapped to any frame in the FrameNet. Those terms have been accommodated in the knowledge base by defining some concepts. Thus, a concept is represented as a collection of terms. For example, there is no entry for 'tiger' in FrameNet and it has been represented through the concept *fearful_animal*. The semantic group *Fearful_Entity* is given in Table 5. In

Table 5. Example of Fearful_Entity semantic group

Fearful_Entity
Terrorist
fearful_animal
Weapon
Catastrophe

this example, the SG *Fearful_Entity* contains three Frames (*Terrorist, Weapon, Catastrophe*) and one concept *fearful_animal*.

An example semantic graph for the EEC describing the killing of people in disease (Killing_by_Disease) is presented in Figure 1.

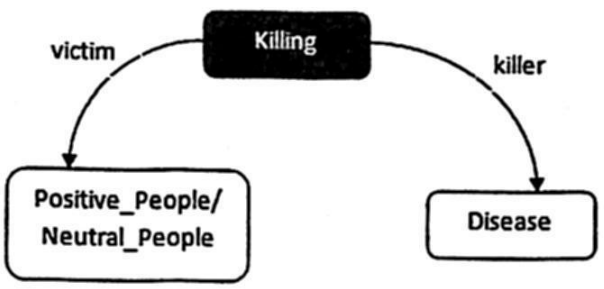

Fig. 1. An example EEC describing the context of killing by disease.

Identifying EEC in Sentence We analyze a sentence to identify the EECs present in it. The semantic parse of the sentence is obtained by means of a semantic parser like SALMANESER. The EEC identification method takes the EEC graphs and the semantic parse graph of a sentence as input and outputs the matched EECs for that particular sentence.

The identification method starts with taking each EEC graph from the EEC knowledge base and tries to fit it into the semantic parse graph. Based on the extent of match, a match score is assigned to the current EEC. The matching process for an EEC graph and a semantic parse graph is depicted in Figure 2. The matching process starts with finding the *pivot* node of the EEC in the

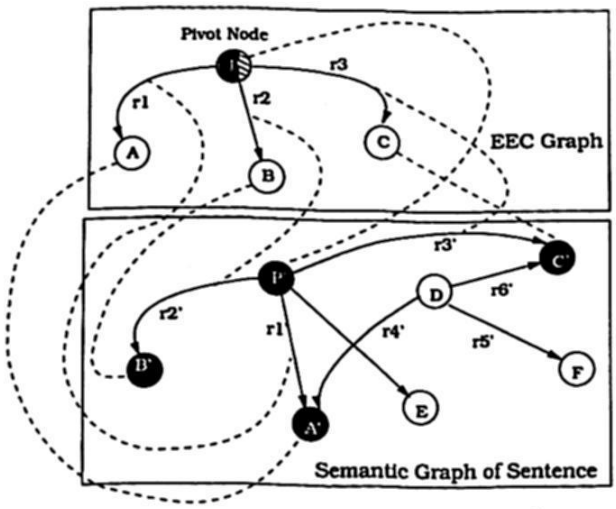

Fig. 2. Illustration of match procedure of an EEC graph and a semantic parse graph

semantic parse graph. Then the relations (an edge and node pair) are probed for matching in semantic parse graph. The match score (m) for an EEC is computed as follows.

$$m = \frac{Number\ of\ relations\ that\ are\ matched}{Total\ number\ of\ relations\ in\ EEC} \tag{2}$$

In the example shown in Figure 2, pivot node P is matched with P' of semantic parse graph. The relations $P - r1 - A$, $P - r2 - B$ and $P - r3 - C$ are matched with $P' - r1' - A'$, $P' - r2' - B'$ and $P' - r3' - C'$ respectively. So, the match score m for this EEC is $\frac{3}{3} = 1$. After computing match scores, the EECs with match scores greater than zero are selected to be the emotion elicitation contexts for the concerned sentence.

By analyzing the corpus, we have defined 62 EECs, some of which are presented in Table 6.

Table 6. Examples of EECs

EEC names		
Reduction_in_Harmful_Entity	Killing_by_Mass_Destruction_Entity	Death_by_Accident
Growth_of_Positive_Entity	Punishing_for_Illegal_Act	Death_by_Catastrophe
Appreciation_of_Entity	Killing_by_Relatives	Performing_Illegal Act
Outbreak_of_Disease	Suffering_from_Disease	Death_by_Suicide

5 Experiments

As we are dealing with fuzzy annotated data, a fuzzy classification framework has been used for developing the emotion classification models. In this section, we present results pertaining to experiments with different feature combinations in FkNN framework.

5.1 Fuzzy k Nearest Neighbor (FkNN) for Emotion Classification

FkNN [13] is a fuzzy extension of popular k nearest neighbor algorithm and it assigns class memberships against a test instance. Let $S = \{s_1, s_2, \ldots s_n\}$ be n labeled training instances and μ_{ij} be the membership of the training instance s_j in i^{th} class. In order to assign membership of a test instance s_t in the i^{th} class, k nearest neighbors of s_t in the training data set are found with a distance measure. The membership of s_t in i^{th} class is determined using equation 3.

$$\mu_i(s_t) = \frac{\sum_{j=1}^{k} \mu_{ij}(1/\|s_t - s_j\|^{\frac{2}{(f-1)}})}{\sum_{j=1}^{k}(1/\|s_t - s_j\|^{\frac{2}{(f-1)}})} \tag{3}$$

The variable f controls the extent to which the distance is weighted while computing a neighbor's contribution to the membership value. Number of nearest neighbors (k) is another parameter in FkNN algorithm.

5.2 Evaluation Measures

As mentioned earlier, the reader perspective emotion analysis is a fuzzy and multi-label classification task. The classification model outputs a membership vector for each test instance where the i^{th} entry μ_i $(0 \leq \mu_i \leq 1)$ is the predicted membership value in the i^{th} class. The evaluation of the fuzzy membership value prediction is performed by measuring Euclidean distance between the predicted and actual membership vector. The evaluation measures those are applicable to multi-label classification task can also be applied here by converting the real valued prediction vector into a binary prediction vector. This kind for conversion is performed by applying α-cut with $\alpha = 0.4$. The evaluation measures used in this study are presented in Table 7.

5.3 Experimental Results

Experiments have been performed with different feature combinations. All the experiments have been performed with $f = 1.5$ and $k = 5$. Results have been reported based on 5-fold cross validation setting for each experiments. Table 8 summarizes the results of emotion classification with different features and their combinations with best results presented in bold face.

When the different features are considered separately, the performance of the emotion classifier with polarity feature (**P**) deteriorated as compared to the

Table 7. Evaluation measures (↑ = Higher the value better the performance, ↓ = lower the value better the performance).

Evaluation Measure Group	Evaluation Measure	Convention
Example based measures [14]	Hamming Loss (HL)	↓
	Partial Match Accuracy (P-Acc)	↑
	Subset Accuracy (S-Acc)	↑
	F1 Measure (F1)	↑
Ranking based measures [15]	One Error (OE)	↓
	Coverage (COV)	↓
	Ranking Loss (RL)	↓
	Average Precision (AVP)	↑
Label based measures [14]	Micro Average F1 (Micro-F1)	↑
	Macro Average F1 (Macro-F1)	↑
Fuzzy membership measure	Euclidean distance (ED)	↓

Table 8. Comparison of features (W = word feature, P = polarity feature, SF = Semantic frame feature, EEC = Emotion elicitation context)

Measure	W	P	SF	EEC	W+P	P+SF	W+SF	W+P+SF	P+EEC
HL	0.136	0.186	0.092	0.063	0.121	0.080	0.099	0.109	**0.051**
P-Acc	0.645	0.562	0.781	0.864	0.705	0.823	0.757	0.728	**0.894**
S-Acc	0.574	0.514	0.692	0.791	0.612	0.740	0.668	0.644	**0.832**
F1	0.679	0.628	0.803	0.899	0.750	0.862	0.800	0.769	**0.925**
OE	0.180	0.213	0.116	**0.059**	0.167	0.090	0.129	0.158	0.061
COV	0.931	1.105	0.719	0.547	0.868	0.656	0.767	0.795	**0.484**
RL	0.163	0.194	0.081	0.053	0.136	0.061	0.104	0.092	**0.041**
AVP	0.850	0.796	0.884	0.925	0.864	0.922	0.889	0.874	**0.957**
Micro-F1	0.688	0.641	0.781	0.827	0.729	0.816	0.750	0.737	**0.877**
Macro-F1	0.615	0.613	0.743	0.818	0.652	0.771	0.681	0.665	**0.857**
ED	0.282	0.302	0.253	0.185	0.284	0.218	0.250	0.260	**0.151**

baseline classifier (using word feature (**W**)) for all the evaluation metrics. This explains how important the terms present in the text are for emotion classification. The use of semantic frames (**SF**) as features improves the performance of emotion classification significantly. This improvement may be attributed to two different transformations over the word feature set.

- *Dimensionality Reduction*: There is a significant reduction in the dimension of semantic frame feature set as compared to word feature set (semantic frame feature dimension = 279 and word feature dimension = 2345).
- *Feature Generalization*: Semantic frame assignment to the terms in the sentences is one generalization technique where conceptually similar terms are grouped into a semantic frame.

On the other hand, a notable improvement have been observed with the use of EEC features. As the contextual information is encoded in EEC feature, it is

more powerful than the semantic frame features. Reduction in dimension with respect to semantic frame feature is also observed in case of EEC feature.

General observations over the feature comparison experiment are as follows.

- The **P+EEC** feature combination performs best in emotion classification with Fuzzy kNN. The **EEC** feature performs closer to **P+EEC** as compared to other feature combinations.
- The polarity feature (**P**) is inefficient than other combinations but whenever coupled with other feature combinations (i.e., **W** vs. **W+P**, **SF** vs. **SF+P**, **W+SF** vs. **W+SF+P** and **EEC** Vs. **P+EEC**), the performance improves. This improvement can be explained with the fact that the polarity feature may help the word or semantic frame based models by classifying the data set into positive and negative category.

6 Conclusions

In this paper, we have presented a fuzzy classification model based on different proposed features in order to perform reader perspective emotion analysis. The problem of reader perspective emotion recognition has been posed as a fuzzy classification problem. We have introduced three new features, namely, polarity, semantic frame and emotion eliciting context based features. Extensive experiments with different feature combinations have been performed and the best performance was achieved with EEC and polarity feature combination.

Acknowledgment

Plaban Kumar Bhowmick is partially supported from projects sponsored by Microsoft Corporation, USA and Media Lab Asia, India.

References

1. Lin, K.H.Y., Yang, C., Chen, H.H.: Emotion classification of online news articles from the reader's perspective. In: Web Intelligence. (2008) 220–226
2. Lin, K.H.Y., Chen, H.H.: Ranking reader emotions using pairwise loss minimization and emotional distribution regression. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Honolulu, Hawaii, Association for Computational Linguistics (2008) 136–144
3. Kozareva, Z., Navarro, B., Vazquez, S., Montoyo, A.: Ua-zbsa: A headline emotion classification through web information. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, Association for Computational Linguistics (2007) 334–337
4. Mihalcea, R., Liu, H.: A corpus-based approach to finding happiness. In: In AAAI 2006 Symposium on Computational Approaches to Analysing Weblogs, AAAI Press (2006) 139–144

5. Alm, C.O., Roth, D., Sproat, R.: Emotions from text: machine learning for text-based emotion prediction. In: HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, Morristown, NJ, USA, Association for Computational Linguistics (2005) 579–586
6. Jung, Y., Park, H., Myaeng, S.H.: A hybrid mood classification approach for blog text. In: PRICAI. (2006) 1099–1103
7. Wu, C.H., Chuang, Z.J., Lin, Y.C.: Emotion recognition from text using semantic labels and separable mixture models. ACM Transactions on Asian Language Information Processing (TALIP) **5** (2006) 165–183
8. Abbasi, A., Chen, H., Thoms, S., Fu, T.: Affect analysis of web forums and blogs using correlation ensembles. IEEE Transactions on Knowledge and Data Engineering **20** (2008) 1168–1180
9. Strapparava, C., Mihalcea, R.: Semeval-2007 task 14: Affective text. In: Proceedings of the 4th International Workshop on the Semantic Evaluations (SemEval 2007), Prague, Czech Republic (2007)
10. Chaumartin, F.R.: Upar7: A knowledge-based system for headline sentiment tagging. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, Association for Computational Linguistics (2007) 422–425
11. Katz, P., Singleton, M., Wicentowski, R.: Swat-mp:the semeval-2007 systems for task 5 and task 14. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, Association for Computational Linguistics (2007) 308–313
12. Fillmore, C.J.: Frames and the semantics of understanding. Quaderni di semantica **6** (1985) 222–254
13. Keller, J., Gray, M., Givens, J.: A fuzzy $k-$nearest neighbor algorithm. IEEE Transactions on Systems Man and Cybernetics. **15** (1985) 580–585
14. Tsoumakas, G., Katakis, I.: Multi label classification: An overview. International Journal of Data Warehouse and Mining **3** (2007) 1–13
15. Zhang, M.l., Zhou, Z.h.: Ml-knn: A lazy learning approach to multi-label learning. Pattern Recognition **40** (2007) 2007

Recognizing Textual Entailment: Experiments with Machine Learning Algorithms and RTE Corpora

Julio J. Castillo
Faculty of Mathematics Astronomy and Physics
Cordoba, Argentina
jotacastillo@gmail.com

Abstract. This paper presents a system that uses machine learning algorithms and a combination of datasets for the task of recognizing textual entailment. The features chosen quantify lexical, syntactic and semantic level matching between text and hypothesis sentences. Additionally, we created a filter which uses a set of heuristics based on Named Entities to detect cases where no entailment was found. We analyze how the different sizes of datasets (RTE1, RTE2, RTE3, RTE4 and RTE5) and classifiers (SVM, AdaBoost, BayesNet, MLP, and Decision Trees) could impact on the final overall performance of the systems. We show that the system performs better than the baseline and the average of the systems from the RTE on both two and three way tasks. We conclude that using RTE3 corpus with Multilayer Perceptron algorithm for both two and three way RTE tasks outperformed any other combination of RTE-s corpus and classifiers in order to predict RTE4 test data.

Keywords: Natural Language Processing, Textual Entailment, Machine Learning, RTE datasets.

1 Introduction

The objective of the Recognizing Textual Entailment Challenge is the task of determining whether or not the meaning of the Hypothesis (H) can be inferred from a text (T). Recently the RTE Challenge has changed to a 3-way task that consists in determining between entailment, contradiction and unknown when there is no information to accept or reject the hypothesis. The traditional two-way distinction between entailment and non-entailment is allowed yet.

In the past RTEs Challenges, machine learning algorithms were widely used for the task of recognizing textual entailment (Marneffe et al., Zanzotto et al., 2007). Thus, in this paper, we tested the most common classifiers that have been used by other researchers in order to provide a common framework of evaluation of ML algorithms (fixing the features) and to show how the development data set could impact over them. We generate a feature vector with the following components for both Text and Hypothesis: Levenshtein distance, a lexical distance based on

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 155-164

Received 29/11/09
Accepted 16/01/10
Final version 09/03/10

Levenshtein, a semantic similarity measure Wordnet based, and the LCS (longest common substring) metric.

We choose only four features in order to learn the development sets. Larger feature sets do not necessarily lead to improved classification performance because they could increase the risk of overfitting the training data. In section 3 we provide a correlation analysis of these features.

The motivation of the input features: Levenshtein distance is motivated by the good results obtained as measure of similarity between two strings. Additionally, we propose a lexical distance based on Levenshtein distance but working to sentence level. We create a metric based in Wordnet in order to capture the semantic similarity between T and H to sentence level. The longest common substring is selected because is easy to implement and provides a good measure for word overlap.

The system produces feature vectors for all possible combinations of the available development data RTE1, RTE2, RTE3 and RTE5. Weka (Witten and Frank, 2000) is used to train classifiers on these feature vectors. We experiment with the following five machine learning algorithms: Support Vector Machine (SVM), AdaBoost (AB), BayesNet (BN), Multilayer Perceptron(MLP), and Decision Trees(DT). The Decision Trees are interesting because we can see what features were selected for the top levels of the trees. SVM, Bayes Net and AdaBoost were selected because they are known for achieving high performance. MLP was used because has achieved high performance in others NLP tasks.

We experiment with various parameters (settings) for the machine learning algorithms including only the results for the best parameters.

For two-way classification task, we use the RTE1, RTE2, RTE3 development sets from Pascal RTE Challenge, RTE5 gold standard and BPI test suite (Boing, 2008). For three-way task we use the RTE1, RTE2, RTE3 development sets from Stanford group, and RTE5 gold standard set. Additionally, we generate the following development sets: RTE1+RTE2, RTE2+RTE3, RTE1+RTE3, RTE1+RTE5, RTE2+RTE5, RTE3+RTE5, RTE2+RTE3+RTE5, RTE1+RTE2+RTE3 and RTE1+RTE2+RTE3+RTE5 in order to train with different corpus and different sizes. In all cases, RTE4 TAC 2008 gold standard dataset was used as test-set.

The remainder of the paper is organized as follows. Section 2 describes the architecture of our system, whereas Section 3 shows results of experimental evaluation and discussion of them. Finally, Section 4 summarizes the conclusions and lines for future work.

2 System description

This section provides an overview of our system that was evaluated in Fourth Pascal RTE Challenge. The system is based on a machine learning approach for recognizing textual entailment. In Figure 1 we present a brief overview of the system.

Using a machine learning approach we tested with different classifiers in order to classify RTE-4 test pairs in three classes: entailment, contradiction or unknown. To deal with RTE4 in a two-way task, we needed to convert this corpus only in two classes: yes, and no. For this purpose, both contradiction and unknown were taken as class no.

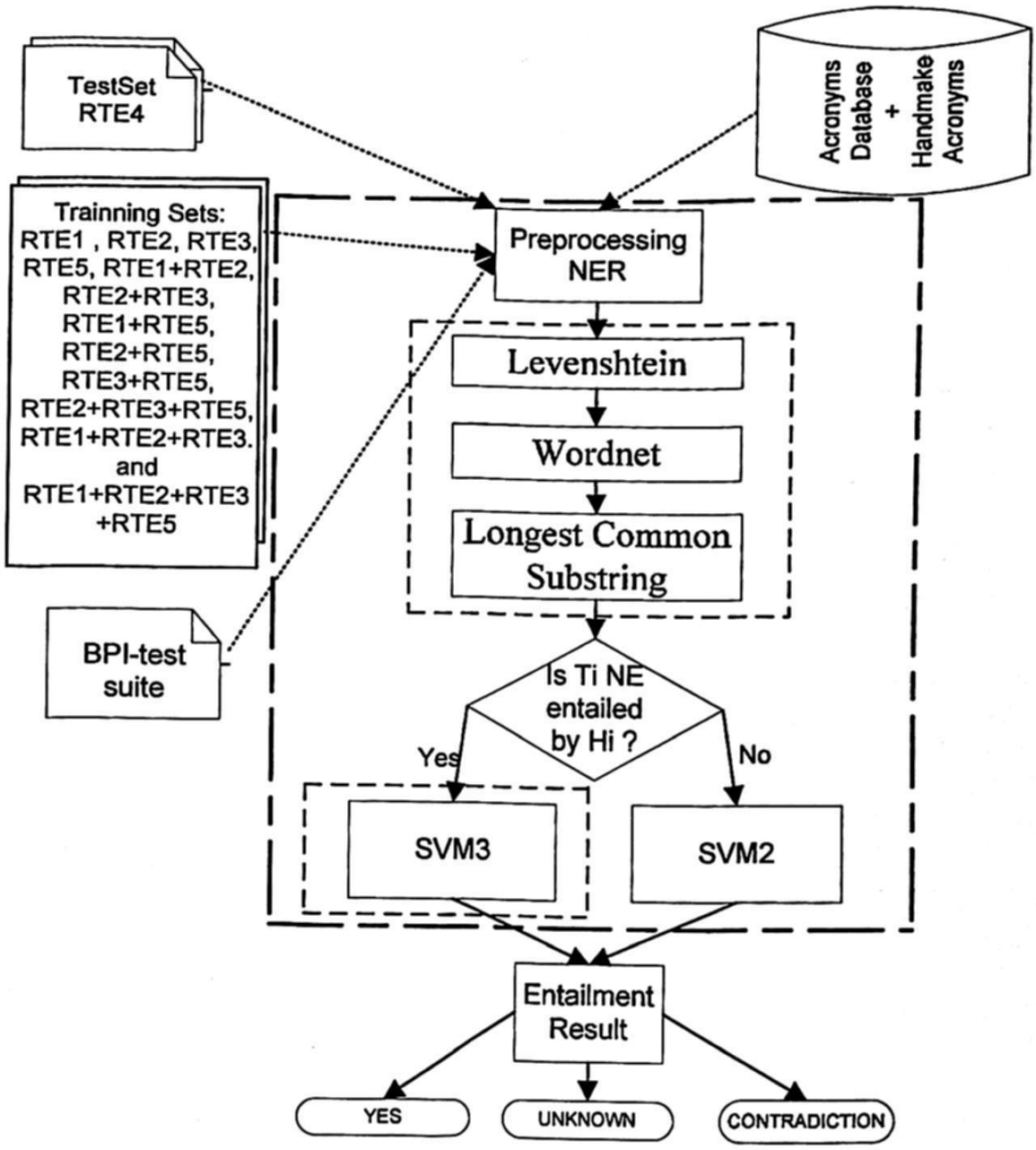

Figure 1. General architecture of our system.

There are two variants to deal with every particular text-hypothesis pair or instance. The first way is directly using four features: (1) the Levenshtein distance between each pair, (2) lexical distance based on Levenshtein, (3) a semantic distance based on WordNet and (4) their Longest Common Substring. The second way is using the "NER- preprocessing module" to determinate whether *non-entailment* is found between text-hypothesis, therefore differing only on the treatment of Named Entities. The Levenshtein distance (Levenshtein, 1966) is computed among the characters in the stemmed Text and Hypothesis strings. The others three features are detailed below.

Text-hypothesis pairs are stemmed with Porter's stemmer and PoS tagged with the tagger in the OpenNLP framework.

2.1 NER Filter

The system applies a filter based on Named Entities. The purpose of the filter is to identify those pairs where the system is sure that *no entailment* relation occurs.

Thus, the NER-preprocessing module performs NER in text-hypothesis pairs applying several heuristics rules to discard when an entailment relation is not found in the pair. In this case, a specialized classifier SVM_2 was trained only with *contradiction* and *unknown* cases of RTE3 corpus and used to classify the pairs between these two classes.

We employed the following set of heuristic rules: for each type of Name Entity (person, organization, location, etc.), if there is a NE of this type occurring in H that does not occur in T, then the pair does not convey an entailment and therefore should be classified as either *contradiction* or *unknown*.

The text-hypothesis pairs are tokenized with the tokenizer of OpenNLP framework and stemmed with Porter's stemmer. We also enhanced this NER-preprocess module by using an acronym database (British Atmospheric Data Centre (BADC)).

The output module was applied to approximately 10 percent of the text-hypothesis pairs of RTE4. The accuracy of the filter evaluated in TAC'08 was 0.71, with 66 cases correctly classified out of 92 where rules applied. An error analysis revealed that misclassified cases were indeed difficult cases, as in the following example (pair 807, RTE4):

Text:
`Large scores of Disney fans had hoped Roy would read the Disneyland Dedication Speech on the theme park's fiftieth birthday next week, which was originally read by Walt on the park's opening day, but Roy had already entered an annual sailing race from Los Angeles to Honolulu.`
Hypothesis:
`Disneyland theme park was built fifty years ago.`

We plan to extend this module so it can also be used to filter cases where an entailment between text and hypothesis can be reliably identified via heuristic rules.

2.2 Lexical Distance

We use the standard Levenshtein distance as a simple measure of how different two text strings are. This distance quantifies the number of changes (character based) to generate one text string from the other. For example, how many changes are necessary in the hypothesis H to obtain the text T. For identical strings, the distance is 0.

Additionally using Levenshtein distance we define a lexical distance and the procedure is the following:

- Each string T and H are divided in a list of tokens.
- The similarity between each pair of tokens in T and H is performed using the Levenshtein distance.

- The string similarity between two lists of tokens is reduced to the problem of "bipartite graph matching", being performed by using the Hungarian algorithm over this bipartite graph. Then, we find the assignment that maximizes the sum of ratings of each token. Note that each graph node is a token of the list.

Finally the final score is calculated by:

$$finalscore = \frac{TotalSim}{Max(Lenght(T), Lenght(H))}$$

Where:
TotalSim is the sum of the similarities with the optimal assignment in the graph.
Length (T) is the number of tokens in T.
Length (H) is the number of tokens in H.

2.3 Wordnet Distance

WordNet is used to calculate the semantic similarity between a T and a H. The following procedure is applied:

1. Word sense disambiguation using the Lesk algorithm (Lesk, 1986), based on Wordnet definitions.
2. A semantic similarity matrix between words in T and H is defined. Words are used only in synonym and hyperonym relationship. The Breadth First Search algorithm is used over these tokens; similarity is calculated using two factors: length of the path and orientation of the path.
3. To obtain the final score, we use matching average.

The semantic similarity between two words (step 2) is computed as:

$$Sim(s,t) = 2 \times \frac{Depth(LCS(s,t))}{Depth(s) + Depth(t)}$$

Where:
s,t are source and target words that we are comparing (s is in H and t is in T).
Depth(s) is the shortest distance from the root node to the current node.
LCS(s,t):is the least common subsume of s and t.
The matching average (step 3) between two sentences X and Y is calculated as follows:

$$MatchingAverage = 2 \times \frac{Match\ (X,Y)}{Length\ (X) + Length\ (Y)}$$

2.4 Longest Common Substring

Given two strings, T of length n and H of length m, the Longest Common Sub-string (LCS) method (Dan, 1999) will find the longest strings which are substrings of both T and H. It is founded by dynamic programming.

$$lcs(T,H) = \frac{Length(MaxComSub(T,H))}{\min(Length(T), Length(H))}$$

In all practical cases, min(Length(T), Length(H)) would be equal to Length(H). All values will be numerical in the [0,1] interval.

3 Experimental Evaluation and Discussion of the Results

We use the following combination of datasets: RTE1, RTE2, RTE3, RTE5(gold-set), BPI, RTE1+RTE2, RTE1+RTE3, RTE2+RTE3, RTE1+RTE5, RTE2+RTE5, RTE3+RTE5, RTE1+RTE2+RTE3, and RTE1+RTE2+RTE3+RTE5 to deal with two-way classification task; and we use the following combination of datasets: RTE1[1], RTE2[1], RTE3[1], RTE5, RTE1+RTE2, RTE2+RTE3, RTE1+RTE3, RTE1+RTE5, RTE2+RTE5, RTE3+RTE5, RTE1+RTE2+RTE3, and RTE1+RTE2+RTE3+RTE5 to deal with three-way classification task. We use the following five classifiers to learn every development set: Support Vector Machine, Ada Boost, Bayes Net, Multilayer Perceptron (MLP) and Decision Tree using the open source WEKA Data Mining Software (Witten & Frank, 2005). In all tables results we show only the accuracy of the best classifier. The RTE4 test set and RTE5-gold set were converted to "RTE4 2-way" and "RTE5-2way" taking *contradiction* and *unknown* pairs as no- entailment in order to assess the system in the two-way task. Tables 1 and 2 shows the results for two-way and three-way task, respectively.

Table 1. Results of two-way classification task.

Training Set	Classifier	Acc %
RTE3	MLP	**58.4%**
RTE3 + RTE5	MLP	57.8%
RTE3 With NER Module	SVM	57.6%
RTE2 + RTE3	MLP	57.5%
RTE1 + RTE2 + RTE3	MLP	57.4%
RTE1+RTE5	MLP	57.2%
RTE5	SVM	57.1%
RTE1 + RTE3	Decision Tree	57.1%
RTE1+RTE2+RTE3+RTE5	MLP	57%
RTE2 + RTE3 + RTE5	Bayes Net	56.9%
RTE2 + RTE5	Decision Tree	56.3%
RTE1 + RTE2	Decision tree	56.2%
RTE2	ADA Boost	55.6%
RTE1	ADA Boost	54.6%
Baselines	-	50%
BPI	BayesNet	49.8%

[1] Data set from Stanford Group.

Table 2. Results of three-way classification task.

Training Set	Classifier	Acc %
RTE3	MLP	**55.4%**
RTE2+RTE3+RTE5	MLP	55.3%
RTE1 + RTE3	MLP	55.1%
RTE1 + RTE5	SVM	54.9%
RTE3 + RTE5	SVM	54.9%
RTE1 + RTE2 + RTE3	MLP	54.8%
RTE1 + RTE2	SVM	54.7%
RTE2	SVM	54.6%
RTE2+RTE3	MLP	54.6%
RTE5	SVM	54.6%
RTE1+RTE2+RTE3+RTE5	SVM	54.6%
RTE2 + RTE5	SVM	54.5%
RTE1	SVM	54%
RTE3-With NER Module	SVM	53.8%
Baseline	-	50%

Here we note that, in both classification tasks (two and three way), using RTE3 instead of RTE2 or RTE1 always achieves better results. Interestingly, the RTE3 training set alone outperforms the results obtained with any other combination of RTE-s datasets, even despite the size of increased corpus. Thus, for training purpose, it seems that any additional datasets to RTE-3 introduces "noise" in the classification task.

(Zanzotto et al, 2007) showed that RTE3 alone could produce higher results that training on RTE3 merged with RTE2 for the two-way task. Thus, it seems that it is not always true that more learning examples increase the accuracy of RTE systems. These experiments provide additional evidence for both classification tasks. However, this claim is still under investigation.

Always the RTE1 dataset yields the worst results, maybe because this dataset has been collected with different text processing applications (QA, IE, IR, SUM, PP and MT), and our system does not have it into account.

In addition, a significant difference in performance of 3.8% and 8.6% was obtained using different corpus, in two-way classification task (with and without the BPI development set, respectively).

In three-way task a slight and not statistical significant difference of 1.4% between the best and worst combination of datasets and classifiers is found. So, it suggests that the combination of dataset and classifier has more impact over 2-way task than over 3-way task.

The best performance of our system was achieved with Multilayer Perceptron classifier with RTE-3 dataset; it was 58.4% and 55.4% of accuracy, for two and three way, respectively. The average difference between the best and the worst classifier for all datasets in two way task was 1.6%, and 2.4% in three-way task.

On the other hand, even if the SVM classifier does not appear as 'favorite' in any classification task, in average SVM is one of the best classifiers.

The performance in all cases was clearly above those baselines. Only using BPI in two-way classification we have obtained a result worst that baseline, and it is because, BPI is syntactically simpler than PASCAL RTE; therefore, it seems to be not enough good training set for machine learning algorithm.

Although the best results were obtained without using the Name Entity Preprocessing module, we believe these results could be enhanced. The accuracy of this module was 71%, but additional analysis of the misclassified instances provide evidence that it could be improved almost up to 80% (e.g.: improving the acronym database, knowledge base information, etc) and thus it could impact positively in overall performance of the system.

With the aim of analyzing the feature-dependency, we calculated the correlation of them. The correlation and causation are connected, because correlation is needed for causation to be proved. The correlation matrix of features is shown below:

Table 3. Correlation matrix of features.

Features	1	2	3	4
1	-	0,8611	0,6490	0,2057
2	0,8611	-	0,6951	0,0358
3	0,6490	0,6951	-	0,1707
4	0,2057	0,0358	0,1707	-

The table shows that features (1) and (2) are strongly correlated, so we experimented eliminating feature (1) to assess the effect on the overall performance over cross validation, and we obtained that accuracy slight decreases in 1%. Similar results are obtained eliminating feature (2).

Finally, we assess our system using cross validation technique with ten folds to every corpus, testing over our five classifiers for both classification tasks. The results are shown in the tables 4 and 5 below.

Table 4. Results obtained with ten folds Cross Validation in three-way task.

Training Set	Classifier	Acc %
RTE3	MLP	**65.5%**
RTE3+RTE5	MLP	61.42%
RTE2 + RTE3	MLP	60.68%
RTE1+RTE2+RTE3	MLP	59.35%
RTE2+RTE3+RTE5	SVM	58.72%
RTE1+RTE2+RTE3+RTE5	SVM	57.74%
RTE5	MLP	57.16%
RTE2	SVM	56.62%
RTE1+RTE2	SVM	55.84%
RTE2+RTE5	MLP	55.28%
RTE1	Decision tree	54.70%
RTE1+RTE5	MLP	53.88%

Table 5. Results obtained with Cross Validation in two-way task.

Training Set	Classifier	Acc %
RTE3	BayesNet	**67.85%**
BPI	BayesNet	64%
RTE1 + RTE2 + RTE3	MLP	63.16%
RTE3+RTE5	BayesNet	63.07%
RTE2+RTE3+RTE5	MLP	61.77%
RTE1+RTE2+RTE3+RTE5	ADA Boost	60.91%
RTE5	SVM	60.33%
RTE2	SVM	60.12%
RTE1+RTE2	MLP	59.79%
RTE2+RTE5	BayesNet	58.78%
RTE1	SVM	57.83%
RTE1+RTE5	SVM	56.93%

The results on test set are worse than those obtained on training set, which is most probably due to overfitting of classifiers and because of the possible difference between these datasets. As before, RTE3 outperforms any other combinations of data sets in ten-fold Cross Validation for both two and three way task.

4 Conclusions

We presented our RTE system which is based on a wide range of machine learning classifiers and datasets. As a conclusion about development sets, we mention that the results performed using RTE3 were very similar to those obtained by the union of the RTE1+ RTE2+RTE3 and RTE3 + RTE5, for both 2-way and 3-way tasks. Thus the claim that using more training material helps seems not to be supported by these experiments.

Additionally, we concluded that the relatively similar performances of RTE3 and RTE3 with NER preprocessing module suggest that further refinements over heuristic rules can achieve better results.

Despite the fact that we did not present here an exhaustive comparison between all available datasets and classifiers, we can conclude that the best combination of RTE-s datasets and classifier chosen for two way task produces more impact that the same combination for three way task, almost for all experiments that we have done. In fact, the use of RTE3 alone improves the performance of our system. Thus, we conclude that RTE3 corpus for both two and three way outperforms any other combination of RTE-s corpus using Multilayer Perceptron classifier.

Future work is oriented to experiment with additional lexical and semantic similarities features and to test the improvements they may yield. Additional work will be focus on improving the performance of our NE preprocessing module.

References

1. Prodromos Malakasiotis and Ion Androutsopoulos. *Learning Textual Entailment using SVMs and String Similarity Measures*. ACL-PASCAL, Prague (2007)
2. Julio Javier Castillo, and Laura Alonso. *An approach using Named Entities for Recognizing Textual Entailment*. TAC 2008, Maryland, USA (2008)
3. M. Lesk. *Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone.* In SIGDOC '86 (1986)
4. Gusfield, Dan. Algorithms on Strings, Trees and Sequences. CUP (1999)
5. V. Levenshtein. *Binary Codes Capable of Correcting Deletions, Insertions and Reversals.* Soviet Physics Doklady, 10:707 (1966)
6. Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco (2005)
7. D. Inkpen, D. Kipp and V. Nastase. *Machine Learning Experiments for Textual Entailment.* RTE2 Challenge, Venice, Italy (2006)
8. F. Zanzotto, Marco Pennacchiotti and Alessandro Moschitti. *Shallow Semantics in Fast Textual Entailment Rule Learners,* RTE3, Prague (2007)
9. Marie-Catherine de Marneffe, Bill MacCartney, Trond Grenager, Daniel Cer, Anna Rafferty and Christopher D. Manning. *Learning to distinguish valid textual entailments.* RTE2 Challenge, Italy (2006)

Text and Speech Generation

Discourse Generation from Formal Specifications Using the Grammatical Framework, GF

Dana Dannélls

NLP Research Unit, Department of Swedish Language,
University of Gothenburg, Sweden
dana.dannells@svenska.gu.se

Abstract. Semantic web ontologies contain structured information that do not have discourse structure embedded in them. Hence, it becomes increasingly hard to devise multilingual texts that humans comprehend. In this paper we show how to generate coherent multilingual texts from formal representations using discourse strategies. We demonstrate how discourse structures are mapped to GF's abstract grammar specifications from which multilingual descriptions of work of art objects are generated automatically.

Key words: MLG, Ontology, Semantic Web, CIDOC-CRM, Cohesion, Discourse strategies, Functional programming.

1 Introduction

During the past few years there has been a tremendous increase in promoting metadata standards to help different organizations and groups such as libraries, museums, biologists, and scientists to store and make their material available to a wide audience through the use of the metadata model RDF (Resource Description Framework) or the Web Ontology Language (OWL) [1, 2]. Web ontology standards offer users direct access to ontology objects; they also provide a good ground for information extraction, retrieval and language generation that can be exploited for producing textual descriptions tailored to museum visitors. These advantages have brought with them new challenges to the Natural Language Generation (NLG) community that is concerned with the process of mapping from some underlying representation of information to a presentation of that information in linguistic form, whether textual or spoken. Because the logical structure of ontologies becomes richer, it becomes increasingly hard to devise appropriate textual presentation in several languages that humans comprehend [3].

In this article we argue that discourse structures are necessary to generate natural language from semantically structured data. This argument is based on our investigations of text cohesive and syntactic phenomena across English, Swedish and Hebrew in comparable texts. The use of a discourse strategy implies that a text is generated by selecting and ordering information out of the underlying domain ontology, a process which provides a resulting text with

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 167-178

Received 19/11/09
Accepted 16/01/10
Final version 09/03/10

fluency and cohesion. It is an approach that relies on the principles drawn from both linguistic and computer science to enable automatic translation of ontology specifications to natural language. We demonstrate how discourse structures are mapped to GF's abstract grammar specifications from which multilingual descriptions of work of art objects are generated automatically. GF is a grammar formalism with several advantages which makes it suitable for this task – we motivate the benefits GF offers for multilingual language generation. In this work, we focus on the cultural heritage domain, employing the ontology codified in the CIDOC Conceptual Reference Model (CRM).

The organization of this paper is as follows. We present some of the principles of cohesive text structure (section 2) and outline the difficulties of following these principles when generating from a domain ontology (section 3). We show how discourse strategies can bridge the gap between formal specifications and natural language and suggest a discourse schema that is characteristic to the cultural heritage domain (section 4). We demonstrate our grammar approach to generating multilingual object descriptions automatically (section 5). We conclude with a summary and provide pointers to future work (section 6).

2 Global and Local Text Structure

Early work on text and context [4] has shown that cultural content is reflected in language in terms of text as linguistic category of genre, or text type. A text type is defined as the concept of Generic Structure Potential (GSP) [5]. According to this definition, any text, either written or spoken, comprises a series of optional and obligatory macro (global) structural elements sequenced in a specific order and that the obligatory elements define the type to which a text belongs. The text type that is expressed here is written for the purpose of describing work of art objects in a museum.

To find the generic structure potential of written object descriptions, we examined a variety of object descriptions, written by four different authors, in varying styles. Our empirical evidence suggest there is a typical generic structure potential for work of art descriptions that has the following semantic groupings:

1. object's title, date of execution, creation place
2. name of the artist (creator), year of birth/death
3. inventory number when entered to the museum, collection name
4. medium, support and dimensions (height, width)
5. subject origin, dating, function, history, condition.

To produce a coherent text structure of an object description the author must follow this semantic specification sequences that convey the macro structure of the text. Apart from the macro structural elements, there is a micro (local) integration among semantic units of the text type that gives the text a unity. These types are reflected in terms of reference types that may serve in making a

text cohesive at the paragraph or embedded discourse level. Some examples of reference types are: conjunction, logical relationships between parts of an argument, consistency of grammatical subject, lexical repetition, consistency of temporal and spatial indicators. Thus local structure is expressed partly through the grammar and partially through the vocabulary.

3 The Realities of a Domain Specific Ontology

The ontology we utilize is the Erlangen CRM. It is an OWL-DL (Description Logic) implementation of The International Committee for Documentation Conceptual Reference Model (CIDOC-CRM) [6].[1] The CIDOC-CRM is an event-centric core domain ontology that is intended to facilitate the integration, mediation and interchange of heterogeneous cultural heritage information and museum documentation.[2] One of the basic principles in the development of the CIDOC CRM has been to have empirical confirmation for the concepts in the model. That is, for each concept there must be evidence from actual data structures widely used. Even though the model was initially based on data structures in museum applications, most of the classes and relationships are surprisingly generic. In the following we use this model to illustrate the limitation imposed by a domain specific ontology on generation where concepts and relationships can not easily be mapped to natural language.

According to the CIDOC-CRM specifications, a museum object is represented as an instance of the concept *E22.Man_Made_Object*, which has several properties including:[3] *P55.has_current_location*, *P108B.has_dimension*, *P45F.consists_of*, *P101F.had_general_use*, *P108B.was_produced_by*. A concrete example of a formal specification (presented in turtle annotation) of the *RestOntheHunt_PE34604* object that was modeled according to the CIDOC Documentation Standards Working Group (DSWG) is given in Figure 1.

Taking the domain ontology structure as point of departure, the information in hand is an unordered set of statements that convey a piece of information about an object. The information the *RestOntheHunt_PE34604* statements convey spans at least four of the semantic sequences that we outline in section 2. To generate a coherent text, some ordering constraints must be imposed upon them. This is in particular important because a statement may map to an addition set of statements about an object, for example the relationship *P108B.was_produced_by* maps to an instance of the concept *E12.Production* that has the following properties: *P14F.carried_out_by*, *P7F.took_place_at*, *P4F.has_time_span*.

[1] The motivation behind the choice of DL is that it allows tractable reasoning and inference; it ensures decidability, i.e. a question about a concept in the ontology can always be answered; it supports the intuition that the model must be clear, unambiguous and machine-processable. These aspects are in particular important in computational setting, where we would like our logic to be processed automatically.

[2] The model was accepted by ISO in 2006 as ISO21127.

[3] *Property* is a synonym for *relationship* that maps between two instances. In this paper we use the term *statement* to refer to a relationship between instances.

```
1.:RestOntheHunt_PE34604
2.        a    :CIDOC5.0.1.rdfsE22.Man-Made_Object;
3. :CIDOC5.0.1.rdfsP30B.custody_transferred_through      :RestOntheHunt_PE34604;
4. :CIDOC5.0.1.rdfsP51F.has_former_or_current_owner      :Museum_Hallwylska;
5. :CIDOC5.0.1.rdfsP51F.has_former_or_current_owner      :Galleria_Spada;
6. :CIDOC5.0.1.rdfsP43F.has_dimension     :width_1.2_m;
7. :CIDOC5.0.1.rdfsP43F.has_dimension     :length_1.54_m;
8. :CIDOC5.0.1.rdfsP50F.has_current_keeper     :Museum_Hallwylska;
9. :CIDOC5.0.1.rdfsP52F.has_current_owner      :Museum_Hallwylska;
10. :CIDOC5.0.1.rdfsP48F.has_preferred_identifier     :PE_34604;
11. :CIDOC5.0.1.rdfsP2F.has_type     :oil_cloth;
12. :CIDOC5.0.1.rdfsP103F.was_intended_for       :RestOntheHunt_function;
13. :CIDOC5.0.1.rdfsP101F.had_as_general_use     :RestOntheHunt_function;
14. :CIDOC5.0.1.rdfsP53F.has_former_or_current_location     :Stockholm;
15. :CIDOC5.0.1.rdfsP53F.has_former_or_current_location     :Rome;
16. :CIDOC5.0.1.rdfsP1F.is_identified_by     :PE_34604;
17. :CIDOC5.0.1.rdfsP1F.is_identified_by     :TA_959a;
18. :CIDOC5.0.1.rdfsP65F.shows_visual_item     :Inscription_of_RestOntheHunt_PE34604;
19. :CIDOC5.0.1.rdfsP45F.consists_of     :linseed_oil;
20. :CIDOC5.0.1.rdfsP45F.consists_of     :canvas_duck;
21. :CIDOC5.0.1.rdfsP62F.depicts         :Bambocciate;
22. :CIDOC5.0.1.rdfsP55F.has_current_location     :Stockholm;
23. :CIDOC5.0.1.rdfsP49F.has_former_or_current_keeper      :Museum_Hallwylska;
24. :CIDOC5.0.1.rdfsP54F.has_current_permanent_location     :Stockholm;
25. :CIDOC5.0.1.rdfsP70B.is_documented_in      :RestOntheHunt.jpg;
26. :CIDOC5.0.1.rdfsP108B.was_produced_by     :Creation_of_RestOntheHunt_PE34604 .
```

Fig. 1. Formal specification of a museum object modeled in the CIDOC-CRM.

4 From Formal Specifications to Coherent Representation

As we pointed out in the previous section, the structure of the ontology is not a good point of departure for producing coherent texts and therefore requires pre-processing. In broad terms this involves taking a set of information elements to be presented to a user and imposing upon this set of elements a structure which provides a resulting text with fluency and cohesion.

Some of the pre-processing steps that have been suggested by previous authors [7, 8] include removing repetitive statements that have the same property and arguments and grouping together similar statements to produce a coherent summary. Although there is a need to select statements that mirror linguistic complexity [9], most authors focus on the semantics of the ontology rather than on the syntactic form of the language. They assume that the ontology structure is appropriate for natural language generation, an assumption which in many cases only applies to English.

In this section we describe the approach we exploit to learn how the ontology statements are realized and combined in natural occurring texts. We perform a domain specific text analysis; texts are studied through text linguistics by which the critic seeks to understand the relationships between sections of the author's discourse.

4.1 Linking Statements to Lexical Units

When text generation proceeds from a formal representation to natural language output, the elements of the representation need to be somehow linked

to lexical items of the language. We examined around 100 object descriptions in English, Swedish and Hebrew and studied how statements are ordered, lexicalised and combined in the discourse. To capture the distribution of discourse entities across text sentences we perform a semantic and syntactic analysis, we assume that our unit of analysis is the traditional sentence, i.e. a main clause with accompanying subordinate and adjunct clauses. Below we exemplify how the ontology statements are mapped to lexical items in the studied texts.[4]
Statements:

1. *P55F. has_current_location* maps between instances of *E22.Man-Made-Object* and instances of *E53.Place* (see line 22, Figure 1)
2. *P52F. has_current_owner* maps between instances of *E22.Man-Made-Object* and instances of *E40. Legal Body* (see line 9, Figure 1)
3. *P82F.at_some_time_within* maps between instances of *E52. Time-Span* and *String* data values.

Text examples:
Eng> The subject made its first appearance [in 1880]$_{P82F}$. It is [now installed]$_{P52F}$ in the Wallace Collection[,]$_{P55F}$ London.

Swe> Först [på 1900 talet]$_{P82F}$ kom den till Sverige och [hänger nu på]$_{P55F}$ Gripsholms slott [i]$_{P52F}$ Statens porträttsamling.

Heb> ha-tmuwnah hegieh larisunah le-Aeretz yisraAel [be-snat 1960]$_{P82F}$. hyA [sayeket le]$_{P52F}$-quwleqitzyah sel Amir bachar [se-nimtzet]$_{P55F}$ be-muwzeyAuwn haAretz be-tel Aabiyb

These text examples exhibit a few local linguistic differences between the languages. In English and Hebrew, the order of the statements is: 3,2,1 while in the Swedish text it is: 3,1,2. It is interesting to note how the domain entities and properties are lexicalized in the different languages. In all three languages the property *P82F.at_some_time_within* is lexicalised with a preposition phrase. On the other hand, the lexicalisation of the property *P55F. has_current_location* differs significantly. Furthermore, in the Swedish text all statements are realized in one single sentence; the statements are combined with a simple syntactic aggregation using the conjunction *och* 'and'. Both in the English and the Hebrew examples, statements 3 and 2 are realized as two sentences which are combined with a referring pronoun, i.e. *it* and *hyA*. When generating natural occuring texts it is important to utilize a generation machinery that supports such syntactic variations. In section 5 we demonstrate how these variation are supported in the GF formalism.

Empirical representations of stereotypical clause structures such as presented above not only provide evidence on how to pair ontology statements with lexical units according to the language specific patterns, but also guide template constructions proceeding according to the organization of the domain semantics.

[4] The transliteration ISO-8859-8 ASCII characters of Hebrew are used to enhance readability.

Table 1. Template specification that governs text structures of a cultural object in a museum.

Name	Template slot
T1	(a) object's title | (b) object's creator | (c) creation date | (d) creation place
T2	(a) creator date of birth | (b) creator date of death
T3	(a) object id | (b) object material | (c) object size
T4	(a) current owner | (b) current location | (c) catalogue date | (d) collection
T5	(a) object's identifier | (b) identified place

4.2 Template Specifications

In section 2 we presented a five stage typical GSP for a work of art object description. To guarantee that the selected statements follow this structure, we defined a sequence of templates describing the discourse structure, this approach was first introduced by [10]. Each sequence in a template consists of slots that correspond to a set of statements in the domain knowledge.

The template specification as whole provides a set of ordering constraints over a pattern of statements in such a way that may yield a fluent and coherent output text. The templates and slots are specified in Table 1.

4.3 A Discourse Schema

A discourse schema is an approach to text structuring through which particular organizing principles for a text are defined. It straddles the border between a domain representation and well-defined structured specification of natural language that can be found through linguistic analysis. This idea is based on the observation that people follow certain standard patterns of discourse organization for different discourse goals in different domains.

Our text analysis has shown certain combinations of statements are more appropriate for the communicative goal of describing a museum object. Following our observations, we defined a discourse schema *Description schema* (see below) consisting of two rhetorical predicates (e.g. Identification–Property and Attributive–Property).[5] The schema encodes communicative goals and structural relations in the analyzed texts. Each rhetorical predicate in the schema is associated with a set of templates (specified in Table 1). The notation used to represent the schema: ',' indicates the mathematical relation *and*, '{}' indicates optionality, '/' indicates alternatives.

Description schema:

Describe–Object – >
  Identification–Property/
  Attributive–Property

Identification–Property – >

[5] The notion of *rhetorical predicates* goes back to Aristotle, who presented predicates as assertions which a speaker can use for persuasive argument.

T1 , {T2 / T3}

Attributive–Property – >
T4 / T5

An example taken from one of the studied texts:

> [T1b]Thomas Sully [T2](1783-1872) painted this half-length [T1a] Portrait of Queen Victoria [T1c] in 1838. The subject is now installed in the [T4d] Wallace Collection, [T4b] London.

The first sentence, corresponding to the rhetorical predicate *Identification–Property*, captures four statements (comprising the following relationships: *P82F.at_some_time_within*, *P14F.carried_out_by*, *P108B.was_produced_by* and *P102. has_title*) that are combined according to local and global text cohesion principles.

5 Domain Dependent Grammar-Based Generation

After the information from the ontology has been selected and organized according to the pre-defined schema, it is translated to abstract grammar specifications. The grammar formalism is the Grammatical Framework (GF) [11], a formalism suited for describing both the semantics and syntax of natural languages. The grammar is based on Martin-Löf's type theory [12] and is particularly oriented towards multilingual grammar development and generation. GF allows the separation of language-specific grammar rules that govern both morphology and syntax while unifying as many lexicalisation rules as possible across languages. With GF it is possible to specify one high-level description of a family of similar languages that can be mapped to several instances of these languages. The grammar has been exploited in many natural language processing applications such as spoken dialogue systems [13], controlled languages [14] and generation [15].

GF distinguishes between abstract syntax and concrete syntax. The abstract syntax is a set of functions (fun) and categories (cat) that can be defined as semantic specifications; the concrete syntax defines the linearization of functions (lin) and categories (lincat) into strings that can be expressed by calling functions in the resource grammar. [6] Each language in the resource grammar has its own module of inflection paradigms that defines the inflection tables of lexical units and a module for specifying the syntactic constructions of the language.

Below we present the abstract and concrete syntax of the rhetorical predicate *Identification–Property* presented in section 4.3.[7] Figure 2 illustrates the abstract syntax tree of our abstract grammar that reflects on the semantics of the domain and that is common for all languages.

---

[6] A resource grammar is a fairly complete linguistic description of a specific language. GF has a resource grammar library that supports 14 languages.

[7] The GF Resource Grammar API can be found at the following URL: `<http://www.grammaticalframework.org/lib/doc/synopsis.html>`.

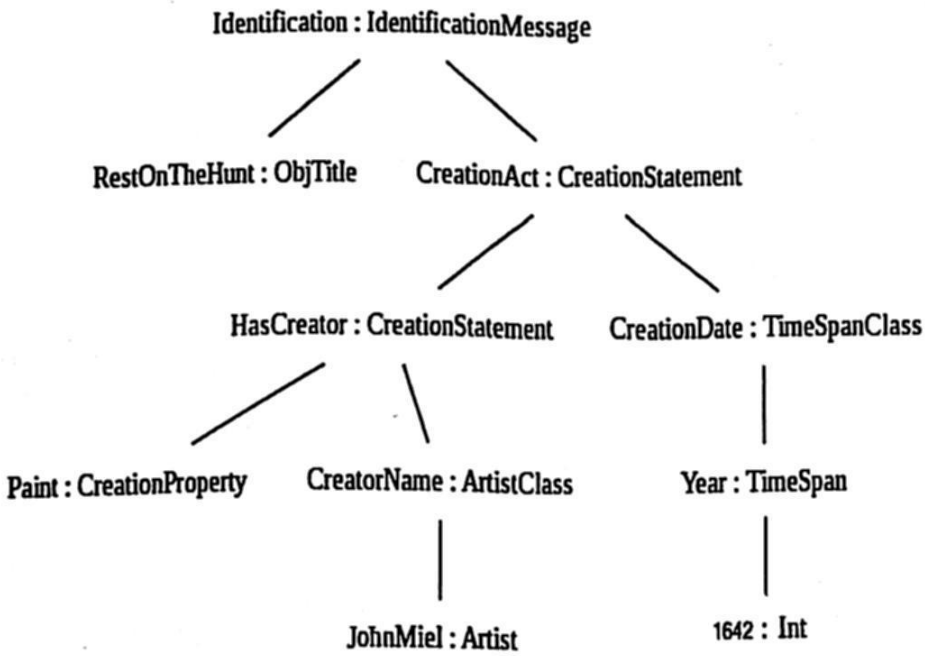

Fig. 2. Abstract syntax tree for *Rest on the Hunt was painted by John Miel in 1642.*

abstract syntax
cat
IdentificationMessage; ObjTitle; CreationProperty; Artist; TimeSpan; CreationStatement;
ArtistClass; TimeSpanClass;
fun
Identification: ObjTitle → CreationStatement → IdentificationMessage;
CreationAct: CreationStatement → TimeSpanClass → CreationStatement;
HasCreator: CreationProperty → ArtistClass → CreationStatement;
CreatorName: Artist → ArtistClass;
CreationDate: TimeSpan → TimeSpanClass;
Year : Int → TimeSpan ;
RestOnTheHunt: ObjTitle;
JohnMiel: Artist;
Paint: CreationProperty;

The abstract specification expresses the semantics of the ontology and is language independent. What makes the abstract syntax in particular appealing in this context is the ability to expand the grammar by simply adding new constants that share both common semantics and syntactic alternations. For example, Beth Levin's [16] English *Performance Verbs* class contains a number of verbs that can be added as constants of type *CreationProperty*, such as *draw* and *produce*, as follows: Paint, Draw, Produce : CreationProperty.

GF offers a way to share similar structures in different languages in one parametrized module called *functor* [17]. In our implementation the common structure of the concrete syntax for English and Swedish is shared in a functor. Since the function *CreationDate* is linearized differently, it is defined separately for each language. This is illustrated below.

incomplete concrete syntax[8]
lincat
IdentificationMessage = S ; TimeSpanClass, ArtistClass = Adv ; TimeSpan = NP ; CreationStatement = VP ; CreationProperty = V2 ; ObjTitle, Artist = PN ;
lin
Identification np vp = mkS pastTense (mkCl (mkNP np) vp);
CreationAct vp compl = mkVP vp compl;
HasCreator v np = (mkVP (passiveVP v) np) ;
CreatorName obj = (mkAdv by8agent_Prep (mkNP obj));
Year y = mkNP (SymbPN y) ;

concrete English syntax
lin CreationDate obj = (mkAdv in_Prep obj);

concrete Swedish syntax
lin CreationDate obj = mkAdv noPrep (mkCN year_N (mkNP obj));

The lexicon is implemented as an interface module which contains *oper* names that are the labels of the record types. It is used by the functor and by each of the language specific lexicons.

interface lexicon
oper
year_N : N;
restOnTheHunt_PN : PN ;
johnMiel_PN : PN ;
paint_V2 : V2 ;

instance English lexicon
oper restOnTheHunt_PN = mkPN ["Rest on the Hunt"]; johnMiel_PN = mkPN "John Miel"; year_N = regN "year"; paint_V2 = mkV2 "paint" ;

instance Swedish lexicon
oper restOnTheHunt_PN = mkPN ["Rastande jägare"]; johnMiel_PN = mkPN "John Miel"; year_N = regN "år"; paint_V2 = mkV2 "måla" ;

[8] The word *incomplete* suggests that the functor is not a complete concrete syntax by itself.

In GF it is possible to built a regular grammar for new languages by using simple record types. In our case we implemented a small application grammar for Hebrew, i.e. *concrete Hebrew* that uses the same abstract syntax as for English and Swedish. In this module functions are linearized as strings where records $\{s : Str\}$ are used as the simplest type.[9] We introduce the parameter type *Gender* with two values: Masc and Fem, these are used in table types to formalize inflection tables. In Hebrew, verb phrases are parameterized over the gender and are therefore stored as an inflection table {s : Gender => Str}; noun phrases have an inherent gender that is stored in a record together with the linearized string {s : Str ; g : Gender}.[10]

concrete Hebrew syntax
lincat
IdentificationMessage, TimeSpan, ArtistClass, TimeSpanClass = {s : Str}; Artist, ObjTitle = {s : Str ; g : Gender}; CreationProperty, CreationStatement = { s : Gender => Str};
lin
Identification np vp = {s = np.s ++ vp.s ! np.g };
CreationAct vp compl = { s = \\g => vp.s ! g ++ compl.s };
HasCreator v obj = { s = \\g => v ! g ++ obj.s};
CreatorName obj = { s = ["al yedey"] ++ obj.s };
CreationDate obj = { s = ["be"] ++ obj.s };
ObjTitle = {s = ["menuhat tzayydym"] ; g = Fem};
JohnMiel = {s = ["guwn miyAe"] ; g = Masc};
Paint = { s = table {Masc => "tzuwyr"; Fem => "tzuwyrah"}};
Param
Gender = Fem | Masc ;

The complete grammar specifications yield the following text, in English, Swedish and Hebrew:

> Eng> Rest on the Hunt was painted by John Miel in 1642. The painting is located in the Hallwyska museum in Stockholm.
> Swe> Rastande jägare blev målad av John Miel år 1642. Tavlan hänger på Hallwyska museet i Stockholm.
> Heb> menuhat tzayydym tzuwyrah 'al yedey guwn miyAel be-1642. htmwnh memukemet be-muwzeyAuwn hallwiska be-stukholm.

This kind of multi-level grammar specification maps non-linguistic information to linguistic representation in a way that supports local and global text variations. For example, in the English and the Hebrew concrete syntax, the sentence complement is realized as a prepositional phrase (signalled by the prepositions *in* and *be*), but in the Swedish sentence, the complement is realized as a noun phrase (signalled by the noun *år*). In the above example this is

[9] The resource grammar for Hebrew is currently under development.

[10] Hebrew has a more complex morphology as the one described here. However, in this implementation we changed the grammar so that it takes only care of gender agreement.

illustrated in the linearization of *CreationDate*. In the Swedish concrete syntax no preposition is used (*noPrep*), and a different NP rule is applied to generate the noun phrase *år 1642*, i.e. CN→ NP → CN. Lexical variations are supported by the grammar as well, for instance, the verb *located* is not a direct translation of the Swedish verb *hänger* 'hang' but the interpretation of the verb in this context implies the same meaning, namely, the painting exists in the Hallwyska museum. The choice of the lexical unit are governed by the semantic structure of the ontology that is reflected in the abstract syntax.

While the functional orientation of isolated sentences of language is supported by GF concrete representations, there are cross-linguistic textual differences that we touched upon in section 4.1 and that are not yet covered in the grammar specifications, i.e. patterns with which cohesive and coherent texts are created. In English, cohesive means comprise conjunction, substitution and ellipsis that can frequently be used to realize a logical relation. In Swedish, cohesive means is often realized as elliptical item, preposition phrase, and/or punctuation. Whereas in Hebrew means of cohesion are realized through the verbal form, usage of ellipsis and conjunctive elements are not common.

6 Conclusion

In this paper we have presented a grammar driven approach for generating object descriptions from formal representations of a domain specific ontology. We illustrated how the lexicons of individual languages pair ontology statements with lexical units which form the backbone of the discourse structure. We demonstrated how schema based discourse structure is mapped to an abstract grammar specification using the domain specific ontology concepts and properties.

We are now in the process of the development of schemata that are being continually modified and evaluated; each rhetorical predicate should capture as many sentence structure variations as possible. A limitation of discourse schemata development is that it requires a lot of human efforts, however once a discourse schema is defined it can automatically be translated to abstract grammar specifications. This method of assembling coherent discourses from basic semantic building blocks will allow any generation system to assemble its texts dynamically, i.e. re-plan portion of its text and communicate successfully.

In the nearest future we intend to extend the grammar to support grouping of rhetorical predicates which requires a certain coverage of linguistic phenomena such as ellipsis, focus, discourse and lexical semantics. The long challenge of this work is in capturing linguistic properties of a language already during the schema development process to guide further development of language independent grammar specifications.

Acknowledgements

The author would like to express her appreciation to Robert Dale for helpful discussions and acknowledge three anonymous readers for commenting on the paper. The GF summer school 2009 and the Centre for Language Technology (CLT) for sponsoring it.

References

1. Schreiber, G., Amin, A., van Assem, M., de Boer, V., Hardman, L., Hildebrand, M., Hollink, L., Huang, Z., Kersen, J., Niet, M., Omelayenko, B., Ossenbruggen, J., Siebes, R., Taekema, J., Wielemaker, J., Wielinga, B.: Multimedian e-culture demonstrator. In Cruz, I.F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L., eds.: International Semantic Web Conference. Volume 4273., Springer (2006) 951–958 E-culture-demonstrator-2006.
2. Bryne, K.: Having triplets – holding cultural data as rdf. In: Proceedings of IACH workshop at ECDL2008 (European Conference on Digital Libraries), Aarhus (2009)
3. Hielkema, F., Mellish, C., Edwards, P.: Evaluating an ontology-driven wysiwym interface. In: Proc. of the Fifth International NLG Conference. (2008)
4. Hasan, R.: Linguistics, language and verbal art. Geelong: Deakin University. (1985)
5. Halliday, M.A., Hasan, R.: Language, Context, and Text: Aspects of Language in a Social-Semiotic Perspective. Oxford: Oxford University Press (1989)
6. Crofts, N., Doerr, M., Gill, T., Stead, S., Stiff, M.: Definition of the CIDOC Conceptual Reference Model. (2005)
7. O'Donnell, M.J., Mellish, C., Oberlander, J., Knott, A.: Ilex: An architecture for a dynamic hypertext generation system. NL Engineering **7** (2001) 225–250
8. Bontcheva, K.: Generating tailored textual summaries from ontologies. In: Second European Semantic Web Conference (ESWC). (2005) 531–545
9. Mellish, C., Pan, J.Z.: Natural language directed inference from ontologies. Artifical Intelligence **172** (2008) 1285–1315
10. McKeown, K.R.: Text generation : using discourse strategies and focus constraints to generate natural language text. Cambridge University Press (1985)
11. Ranta, A.: Grammatical framework, a type-theoretical grammar formalism. Journal of Functional Programming **14** (2004) 145–189
12. Martin-Löf, P.: Intuitionistic type theory. Bibliopolis, Napoli (1984)
13. Ljunglöf, P., Larsson, S.: A grammar formalism for specifying isu-based dialogue systems. In: Advances in Natural Language Processing, 6th International Conference, GoTAL 2008, Gothenburg, Sweden. Volume 5221 of Lecture Notes in Computer Science., Springer (2008) 303–314
14. Khegai, J., Nordström, B., Ranta, A.: Multilingual syntax editing in gf. In Processing, I.T., (CICLing-2003), C.L., eds.: LNCS 2588, Mexico, Springer (2003) 453–464
15. Johannisson, K.: Formal and Informal Software Specifications. PhD thesis, Chalmers University of Technology (2005)
16. Levin, B.: English Verb Classes and Alternations: A Preliminary Investigation. University of Chicago Press, Chicago (1993)
17. Ranta, A.: The gf resource grammar library. Linguistic Issues in Language Technology (LiLT) (2009)

An Improved Indonesian Grapheme-to-Phoneme Conversion Using Statistic and Linguistic Information

Agus Hartoyo, Suyanto

Faculty of Informatics - IT Telkom, Jl. Telekomunikasi No. 1 Terusan Buah Batu
Bandung, West Java, Indonesia
truegushar@yahoo.co.id, suy@ittelkom.ac.id

Abstract. This paper focuses on IG-tree + best-guess strategy as a model to develop Indonesian grapheme-to-phoneme conversion (IndoG2P). The model is basically a decision-tree structure built based on a training set. It is constructed using a concept of information gain (IG) in weighing the relative importance of attributes, and equipped with the best-guess strategy in classifying the new instances. It is also leveraged with two new features added to its pre-existing structure for improvement. The first feature is a pruning mechanism to minimize the IG-tree dimension and to improve its generalization ability. The second one is a homograph handler using a text-categorization method to handle its special case of a few sets of words which are exactly the same in spelling representations but different each other in phonetic representations. Computer simulation showed that the complete model performs well. The two additional features gave expected benefits.

Keywords: Indonesian grapheme-to-phoneme conversion, IG-tree, best-guess strategy, pruning mechanism, homograph handler.

1 Introduction

Many methods of data driven approach was proposed to solve grapheme-to-phoneme (G2P) conversion problem, such as instance-based learning, artificial neural networks, and decision-tree. In [7], it was stated that an IG-tree + best-guess strategy has high performance. It compresses a given training set into an interpretable model. In this research, the method is adopted to develop a new model for Indonesian G2P (IndoG2P). In the new model, two new features for improvement are added: a pruning mechanism using statistic information and a homograph handler based on some linguistic information provided by a linguist.

According to the fact that the model is a lossless compression structure, which means that it stores all data including those of outliers into rules, a pruning mechanism is proposed to prune some rules accommodating outliers. Hence, the model is expected to increase its generalization, but decrease its size.

Furthermore, the model does not handle homograph problems. The letter-based inspection mechanism performed letter by letter internally in a word cannot handle a few sets of words which are exactly the same in spelling representations but different

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 179-190

Received 23/11/09
Accepted 16/01/10
Final version 12/03/10

in phonetic representations. In order to solve this problem, the system should perform an inspection mechanism for wider context. It is clear that the problem is actually only how to recognize the topic of the surrounding text (sentence, paragraph, or passage) which is known as the problem of text categorization. Since [8] stated that the centroid-based classifier for text categorization significantly outperforms other classifiers on a wide range of data sets, the classifier is adopted to solve homograph problem in this research.

2 IndoG2P

The IndoG2P system is designed to use both statistic and linguistic information. This design is expected to solve some different problems in G2P.

2.1 The Phonetic Alphabets

This research uses IPA (International Phonetic Association) Indonesian alphabet system to symbolize pronunciations at the phonetic side of its dataset. As explained in [1], 6 vowels and 22 consonants compose the alphabet system as listed in table 1.

Table 1. The IPA Indonesian alphabets

Phoneme	Category	Sample Word	Phoneme	Category	Sample Word
a	vowel	/akan/	l	consonant	/lama/
e	vowel	/sore/	m	consonant	/makan/
ə	vowel	/ənam/	n	consonant	/nakal/
i	vowel	/ini/	p	consonant	/pintu/
o	vowel	/toko/	r	consonant	/raja/
u	vowel	/baru/	s	consonant	/sama/
b	consonant	/tembakan/	t	consonant	/timpa/
c	consonant	/cari/	w	consonant	/waktu/
d	consonant	/duta/	x	consonant	/axir/
f	consonant	/faksin/	y	consonant	/yakin/
g	consonant	/gula/	z	consonant	/zat/
h	consonant	/hari/	š	consonant	/mašarakat/
j	consonant	/juga/	ŋ	consonant	/təmpuru-ŋ/
k	consonant	/kaki/	ň	consonant	/ňañian/

2.2 Datasets

The system involves two datasets: 1) IndoG2P datasets used to train the system in building the IG-tree model validate the rules during the IG-tree pruning process, and test the IG-tree classifier as IndoG2P conversion system; and 2) Homograph dataset used to train the centroid-based classifiers and test them as homograph handler.

IndoG2P Datasets. "Given a written or spelled word in Indonesian, the system should output how the word is pronounced" is the main problem IndoG2P conversion must cope with. So, this dataset should be able to give examples for the learning system about how words in Indonesian are spelled and then pronounced. It is simple to understand that the dataset can be a table with two attributes with each record demonstrating spelling and pronunciation of a word: spelling transcription of a word on the first attribute and phonetic transcription of the same word on the second one. The format of the dataset is shown in Table 2.

Table 2. Format of the IndoG2P dataset.

Graphemic transcription	Phonetic transcription
malang	mala-ŋ
tembakan	tembakan
tempurung	təmpuru-ŋ
tempatmu	təmpatmu

The learning mechanism requires that type of the relation between spelling symbols and their corresponding phonetic symbols is one-to-one mapping. It seems to be no problem in case the spelling and phonetic transcriptions of a word is basically in the same length as shown in word "tembakan" (meaning "a shot") and "tempatmu" (meaning "your place"). In case the spelling and phonetic transcriptions of a word basically differ in length, [7] suggests to perform an alignment mechanism as shown in word "malang" (meaning "poor" or "unfortunate") and "tempurung" (meaning "shell" or "skull") by inserting phoneme null '-' at a certain position in the phonetic transcription in such way that (i) every single spelling symbol is mapped to a single phonetic symbol; (ii) its grapheme-to-phoneme mapping is viable (i.e. that they can be motivated intuitively or linguistically); (iii) the combination of all mappings within the alignment has a maximal probability; and (iv) it is consistent with alignments of other similar words.

Table 3. The IndoG2P datasets.

Dataset	Number of instances	Percentage
Training set	5,455	80%
Validation set	679	10%
Test set	679	10%

In this research, IndoG2P dataset is developed from a corpus of words collected from articles published by an Indonesian newspaper, and its grapheme-phoneme pairs' correctness was validated by a professional Indonesian linguist. The dataset consist of 6,791 distinct instances which are then randomly divided into three subsets those are a training set to train the system in building the IG-tree model, a validation set to validate the rules during the IG-tree pruning process, and a test set to test the IG-tree classifier. Proportions of the three subsets are illustrated by Table 3.

Homograph Datasets. These are datasets used in the module of homograph handler. The datasets are composed by texts as their instances. In this module, a particular dataset is provided for a particular homograph word. Any text in a dataset for a homograph word must: 1) contains at least one occurrence of the related homograph word; 2) be composed with relevant sentences; and 3) be labeled with the category representing phonetic representation of the ambiguous graphemes. Our real-world datasets are in this research composed with used texts taken from many articles in the Internet. Referencing to the list of Indonesian homograph words shown in Table 5, we provide 5 datasets for 5 homograph words as follows.

Table 4. Homograph datasets

Homograph word	Training set	Test set
apel	80	20
penanya	14	5
sedan	48	12
mental	40	10
tahu	48	12

2.3 IG-Tree + Best-Guess Strategy

As a lossless compression structure, the origin (without pruning mechanism) IG-tree stores in a compressed format the complete grapheme-to-phoneme knowledge of all words provided by the training set. In this system, compression doesn't only mean the decrease of the model's size, but it means generalization as well. As explained in [7], the generated rules can be seen as optimized, generalized lexical lookup. Words spelled similarly are pronounced similarly since the system's reasoning is based on analogy in the overall correspondence of the grapheme-to-phoneme patterns. The system automatically learns parts of words on which similarity matching can be safely performed. At the end of the learning phase a generated rule actually corresponds to a grapheme with a minimal context which disambiguate mapping of the grapheme into a certain phoneme.

For an illustration, see how the system determines the phonetic representation for grapheme <e> in <tempayan> (meaning "large water jar") when dataset shown in Table 2 is given as its learning materials. Based on the learning materials the system finds that grapheme <e> has two probable phonemic representations, those are /e/ and /ə/. Both maximal subword chunk <tembakan> and <tempatmu> actually disambiguate the <e> mapping patterns, in the meaning that the context surroundings <e> in chunk <tembakan> certainly leads its <e> to be mapped to /e/ in the same way as that in chunk <tempatmu> certainly leads its <e> to be mapped to /ə/.

However the sub-word chunk does not represent minimal context disambiguating the mapping patterns. In contrast, subword chunk represents a smaller context but it is ambiguous since this chunk belongs to some words with different <e>'s phonetic representation. Hence, this system during the learning phase will look for more contextual information and finally find that subword chunk <temb> represents the minimal context disambiguating the focus grapheme <e> to be pronounced as /e/

and <temp> represents the minimal context disambiguating the focus grapheme <e> to be pronounced as /ə/. So, when the system is requested to determine what a certain grapheme in a given word should be pronounced, it will find a mapping pattern on the focus grapheme and matching context, and then get the phonetic label led by the pattern as the answer. In our case since the given word <tempayan> on focus grapheme <e> matches with context represented by subword chunk <temp>, it suggests the system to map the focus grapheme to phoneme /ə/ instead of /e/. When other words such "ditempati" (meaning "being inhabited") and "tempatku" (meaning "my place") are given, the generalization ability of the system is shown, as the same rule covers these cases as well.

Dataset Transformation. On the lowest level, instead of running word by word, IndoG2P conversion actually runs letter by letter. If our problem is considered as a classification problem, given an unknown instance with attributes of a focus grapheme and its context graphemes, IndoG2P is a classification task responsible to label the instance with a phonetic representation. This awareness suggests us to transform the IndoG2P dataset discussed before — a word-by-word dataset, we can say — to its new format of letter-by-letter dataset. The basic idea of the transformation's algorithm is consecutively locating each grapheme (occurred in a words) as the focus / target grapheme and ensuring that when a grapheme is located as focus, other graphemes occurred in the same word are simultaneously located on their appropriate context position. As the number of records belonging to word-by-word dataset is the number of words itself, the number of records belonging to letter-by-letter dataset is the total number of letters occurred in whole words. This transformation is illustrated in Fig. 1.

Graphemictranscription	Phonemic transcription
kamper	kampər
tembak	tembak

L7	L6	L5	L4	L3	L2	L1	F	R1	R2	R3	R4	R5	R6	R7	Phonemic label
.	.	.	.	.	.	^	k	a	m	p	e	r	^	.	k
.	.	.	.	.	^	k	a	m	p	e	r	^	.	.	a
.	.	.	.	^	k	a	m	p	e	r	^	.	.	.	m
.	.	.	^	k	a	m	p	e	r	^	.	.	.	.	p
.	.	^	k	a	m	p	e	r	^	.	.	.	.	.	ə
.	^	k	a	m	p	e	r	^	.	.	.	.	.	.	r
.	.	.	.	.	.	^	t	e	m	b	a	k	^	.	t
.	.	.	.	.	^	t	e	m	b	a	k	^	.	.	e
.	.	.	.	^	t	e	m	b	a	k	^	.	.	.	m
.	.	.	^	t	e	m	b	a	k	^	.	.	.	.	b
.	.	^	t	e	m	b	a	k	^	.	.	.	.	.	a
.	^	t	e	m	b	a	k	^	.	.	.	.	.	.	k>

Fig. 1. Dataset transformation.

As shown in Fig. 1, we provided 14 context graphemes surrounding the focus grapheme; those are 7 for each of right and left side (as R1 represents the first context on the right, L1 represents the first context on the left, and so on). The width of context provided in the dataset should be able to accommodate the context expansion (More about context expansion is discussed in the next section.) performed during the learning process. It shouldn't be too narrow as disambiguation expansion point cannot

be reached for patterns with long condition. It shouldn't be too wide either as the system will be too space-consuming. Our determination for 7 bidirectional surrounding graphemes as the width of the context is based on the result of our early investigation stating that phonetic mapping of a grapheme in any ordinary Indonesian words is ambiguous until at most 5 steps to right and/or left side. We gave 2 extra steps to anticipate extraordinary materials in the real dataset.

IG-Tree Construction. The IG-tree is a compression format of context-sensitive rules. Each path in the decision tree represents a rule and is started by a node representing a focus grapheme to be mapped to a phoneme; and the consecutive node represents the consecutive context. Information gain (IG), a computational metric based on information theory, is used to determine the order of context's expansion. Higher information gain of an attribute theoretically reflects less randomness or impurity of partitions resulted by partitioning on the attribute, while in our case it indicates more importance of the attribute in disambiguating the grapheme-to-phoneme mapping. The result of the information gain computation on our IndoG2P dataset (in its letter-by-letter format) for each attribute is described in the graphic in Fig.2.

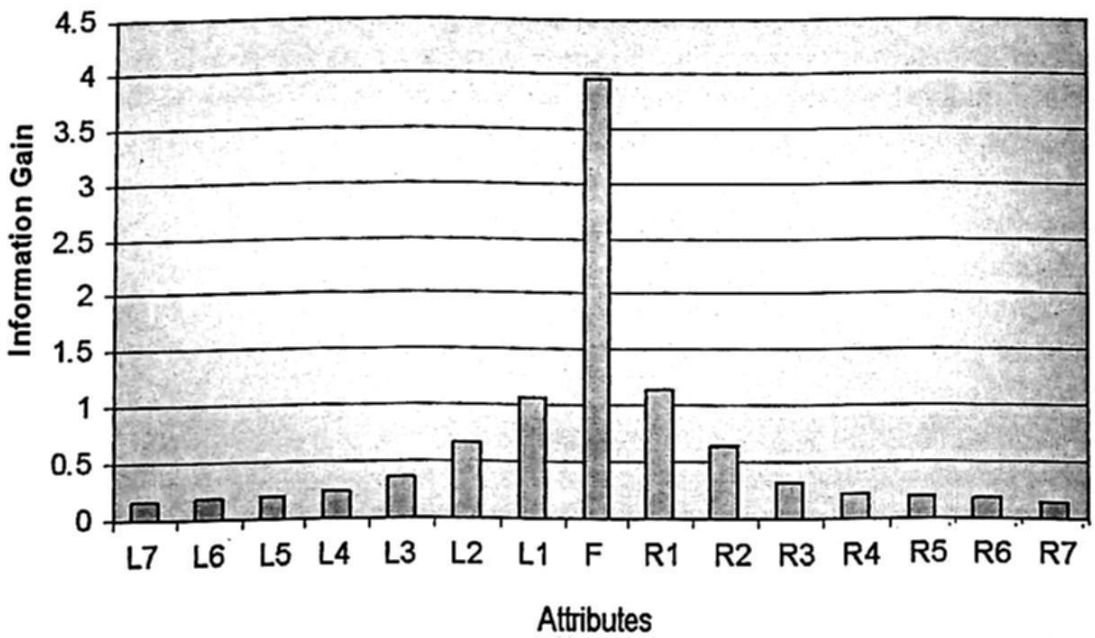

Fig. 2. Information gain for attributes in IndoG2P dataset.

Note that an attribute with the highest importance in disambiguating the grapheme-to-phoneme mapping is the attribute F (the focus grapheme). This fact is consistent with what we have stated above that the focus grapheme is located as the starting node on every path in the IG-tree. Furthermore, the graphic clearly shows us that the information gain is getting smaller as the context position is getting further from the focus grapheme. The system sorts the attributes based on their information gain values, records the order, and uses it to determine the next attribute to which the context inspection should expand during the learning process. In our case, based on

the result of our computation, we got this attribute order: F – R1 – L1 – L2 – R2 – L3 – R3 – L4 – R4 – R5 – L5 – L6 – R6 – L7 – R7.

The construction of IG-tree, just similar with those of some standard decision tree structures such as ID3, C4.5, and CART, can be practically done in such a top-down divide-and-conquer manner. Performed on our letter-by-letter training set, the basic algorithm of the decision tree construction is greedy and can be expressed recursively as follows.

Base: If the current node is pure, i.e. instances reaching that node are totally of the same class, return the node as a leaf node labeled with that class. Stop developing that part of the tree;
Recurrence: Otherwise, i.e. instances reaching the current node differ in class labels, use a splitting attribute to divide the instances in such way that it splits up the example set into subsets, one for each value of the attribute. Apply this procedure to each subset.

We must add information to this recursive algorithm specifically for our case that the splitting attribute, mentioned in the recurrence, at anytime is governed by the attribute order we have discussed earlier. The order is applied constantly for every path in the decision tree.

Another feature belonging to IG-tree is the statistic recording mechanism performed on every non-leaf node. This mechanism records the phonetic label's statistic of all instances reaching each node. This statistic will be employed to perform best-guess strategy and pruning mechanism.

To illustrate how the model exactly is, we will use dataset shown in Table 2 as our study case. As the dataset is still in word-by-word version, our workflow demands it to be transformed to its letter-by-letter format. The IG-tree construction using the procedure we have discussed above is then performed on the later format of dataset. Fig. 3 illustrates a part, i.e. the part of grapheme <e> mapping paths, of the IG-tree constructed.

Retrieving Phonetic Label in IG-Tree. Given a focus grapheme and its context graphemes, the phonetic label of the focus grapheme is basically retrieved by tracing through a proper path in the IG-tree in direction to its leaf and returning the label of the leaf as the phonetic label requested. How we get phoneme /ə/ as the phonetic label of grapheme <e> in word <tempayan> will be more clearly discussed in this section. Based on the attribute order we have discussed above, the tracing will start with node labeled <e> as the focus grapheme. The node labeled with the first grapheme on the right of <e> in <tempayan>, i.e. <m>, is the second node accessed in the path. The next taken node is that labeled <t>, the first grapheme on the left of <e>; continued with node labeled <-> as the second "grapheme" on the left of <e>. It is the end of the tracing when node labeled <p>, the second grapheme on the right of <e>, is accessed, since this node is a leaf node. Thus, label /ə/ is retrieved as the phonetic label corresponding with grapheme <e> in word <tempayan>. In the similar way, phoneme /e/ can be retrieved as the phonetic label for grapheme <e> in word <tembaklah>. However, in this case our tracing will fail in reaching any leaf node when our word is, for instance, <teman> with focus on grapheme <e>. Note that the tracing gets stuck

on node labeled <-> since this node have no child labeled <a>. Such problem is in our research solved with best-guess strategy.

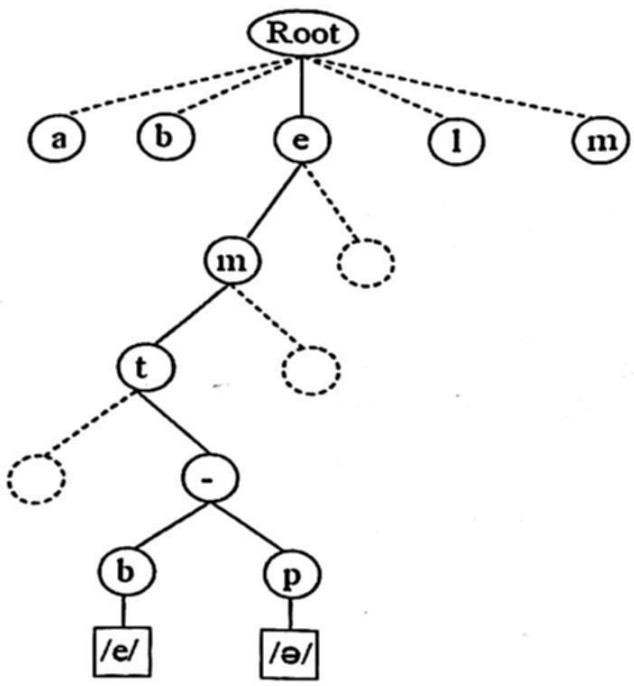

Fig. 3. The IG-tree constructed on dataset in Table 2 with stressing in <e> mapping

Best-Guess Strategy. This is a strategy employed by the system to avoid stumped mapping and to increase its generalization ability. The strategy is performed when a tracing to retrieve a phonetic label cannot reach any leaf node. When the tracing gets stuck on a node, the system will "guess" the phonetic label with the most probable label on that node. The most probable label is computed using statistic information stored on that node. The most frequent phonetic label on records affiliated with the node will be returned as the "guessed" label. In case that there are more-than-one phonetic labels are majorities, a random selection among the labels is performed.

2.4 Pruning Mechanism

When a decision tree is built, many of its branches reflect anomalies due to outliers in the learning materials. The pruning mechanism is proposed to address this problem of overfitting the data. This is performed after the IG-tree is initially grown to its entirety. Pruning is done by replacing a subtree with a new leaf node labeled with the subtree's majority phonetic label. The new tree is then validated using validation set. If the pruning step decreases the generalization accuracy, the previous subtree will be retained; otherwise, the subtree will be permanently removed by the new leaf node. In the case that the accuracy is constant, it was stated that for the same performance the more concise model is the better. This procedure is then performed on all subtrees in a bottom up fashion.

2.5 Centroid-based Text Categorization

Indonesian has some homograph words as shown in Table 5. Daelemans et al in [7] said that their research fails in handling such words in their G2P conversion system. We redeem their failure by proposing text categorization approach using the centroid-based classifier to cope with the problem.

This approach of text categorization and the main system of the IndoG2P conversion using IG-tree actually work on different level. While the IG-tree works on the level of letter with context inspection mechanism internally in its *containing* word, the approach of text categorization works on the level of text with computation on some aspects of its *contained* words / terms. It implies that this proposed approach is applicable on the system with text inputs, not only word inputs.

How centroid-based text categorization copes with the homograph problem is shortly explained as follows. In this approach each homograph word is treated as a particular problem to solve. It demands that a particular learning step, surely with a particular dataset, is performed for each homograph word. Furthermore in this approach, the text on each instance is represented as a vector in term space with TF-IDF computation for each of its dimension. A centroid model of each category is then constructed as average representation of all instances labeled with that category. When an unlabeled instance is given, the classifier computes similarity between vector representing the instance and vector representing centroid of each category. The category with the most similar centroid is output as the label for the new instance. Thus, with the centroid models constructed in the learning process, IndoG2P can correctly map the ambiguous grapheme in a given homograph word surrounded by a particular text, to its phonetic representation.

Table 5. The list of some Indonesian homograph words.

Homograph word	Ambiguous grapheme	Phonetic representation
<apel>	<e>	/apəl/,/apel/
<penanya>	<e>	/pənaña/,/penaña/
<memerah>, <pemerah>, <pemerahan>	<e>	/məmerah/, /pəmerah/, /pəmeraĥan/, /məmərah/, /pəmərah/, /pəməraĥan/
<seri>	<e>	/səri/,/seri/
<semi>	<e>	/səmi/,/semi/
<sedan>	<e>	/sedan/,/sədan/
<mental>	<e>	/mental, /məntal/
<seret>	<e>	/seret/, /sərət/
<serak>	<e>	/sərak/, /serak/
<tahu>	<h>	/taĥu/, /tahu/
<gulai>	<i>	/gulay/,/gulai/

3 Evaluation and Results

Although both modules are practically not detachable, the IG-tree construction and the homograph handling substantially address different level of problem. Hence, their datasets are different as discussed before. So, we divided this section into two parts, each for a particular module.

3.1 IG-Tree Construction

We will see in this part the model improvements due to pruning. The performance of the final model is then evaluated using two accuracy metrics; those are accuracy-per-phoneme and accuracy-per-word. Since on the lowest level the mapping is done letter-per-letter, the computation for accuracy-per-phoneme is done to accommodate evaluation on this level. On the other hand, accuracy-per-word is more interpretable in the sense that the communication using linguistic tools in the real world is word-based, not letter-based. Technically note that accuracy-per-word must be less than or equal to accuracy-per-phoneme since to get a word correctly-mapped, its contained letters are needed to be all true, but in the opposite, one mispronounced letter is sufficient to lead the word in which it occurs to be false.

The resulted IG-tree has 2,112 leaf nodes. The number of the leaves in this section trivially represents the model's dimension. The pruning mechanism is then performed on the model to decrease the model's dimension and increase the accuracy as shown in Table 6. After 554 iterations, dimension of the final IG-tree is 57% smaller than that of the original and its accuracy for validation set is better.

Table 6. Pruning and its improvements in dimensional and accuracy for validation set.

Pruning iteration	Number of leaves	Phoneme accuracy	Word accuracy	Mean accuracy	Mean error
0	2,112	99.19	93.96	96.57	3.43
1	2,111	99.19	93.96	96.57	3.43
2	2,096	99.19	93.96	96.57	3.43
3	2,074	99.19	93.96	96.57	3.43
110	1,871	99.23	94.26	96.74	3.26
221	1,658	99.23	94.26	96.74	3.26
332	1,496	99.30	94.70	97.00	3.00
443	1,315	99.36	95.14	97.25	2.75
554	908	99.38	95.29	97.33	2.67

The final IG-tree was then tested using our test set. The test gave us result of 99.01% for phoneme accuracy and 92.42 % for word accuracy. Unfortunately, we did not find any similar system working on the same language to compare with. However, this result is much better than those of similar researches on other languages published in [3], [4], [5], [6], [8], [9], [10], and [11]. It seems that the method we used and the language we worked on are factors contributing the most for this result. It is stated in [7] that the high performance of IG-tree in G2P conversion suggests

overstatement on previous knowledge-based approaches as well as more computationally expensive learning approaches. Moreover, in this research we improved the original method with new features which increase its performance. About the language we worked on, it is clear that linguistically Indonesian has much simpler phonetic rules than other languages, like English, Dutch, and French. These simple Indonesian linguistic patterns seem to be easily caught during the learning process, so the system could perform better.

Another aspect of IG-tree we want to stress on is its aspect of interpretability. As it was constructed in decision tree structure whose characteristic is descriptive as well as predictive, IG-tree gives us reason for each of its prediction. The model "teaches" us the complete detailed "lesson" about how to pronounce letters in Indonesian words. In practical level, the comprehensive rules exposed in the model constructed even perhaps can help Indonesian linguists codifying Indonesian pronunciation standards.

3.2 Homograph Handler

In this part of research we conducted experiments on five cases for five homograph words as mentioned in Table 4. However, with small number of training instances and test instances, in every single case we got so surprisingly perfect accuracy of the model built during its learning process that extremely no mistake was made by its centroid-based classifier in predicting the category of some new homograph-containing texts in its particular test set.

The well-designed features of the centroid-based classifiers are factors playing the main role in the perfect performances of the models in the five cases. The other factor is the well-prepared dataset used both in each learning and test process. In spite of their small sizes, the datasets are noise-free, discriminating, and balance.

4 Conclusion

The high performance of the system implies that the IG-tree + best-guess strategy is a powerful, language-independent, reasoning-based method well-designed to cope with grapheme-to-phoneme conversion problem nowadays. The result is furthermore contributed also by the characteristic of Indonesian itself whose pronunciation rules are relatively easy for the learning system to catch. The proposed pruning mechanism successfully improves the system's performance by significantly reducing the dimension of the model and increasing its generalization ability. The high interpretability of the model developed in this work must be considered as one aspect of its high performance as well. In addition, the homograph problems, totally unsolved or unsatisfying-solved in previous works, can be handled very well with the proposed centroid-based classifiers.

Acknowledgments

We would like to thank to Mrs. Dyas Puspandari as an Indonesian linguist and all colleagues at IT Telkom for the kind and useful suggestions.

References

1. Alwi, Hasan et al. 2003. *Tata Bahasa Baku Bahasa Indonesia*. Edisi Ketiga. Jakarta: Balai Pustaka
2. Asian, J., Williams, H. E., Tahaghoghi, S. M. M.: Stemming Indonesian. In: Proceedings of the Twenty-eighth Australasian conference on Computer Science, Newcastle, Australia, pp. 307--314 (2005)
3. Bouma, G.: A finite state and data-oriented method for grapheme-to-phoneme conversion. In: Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference, Seattle, Washington, pp. 303-310 (2000)
4. Caseiro, D., Trancoso, I., Oliveira, L., Viana, C.: Grapheme-to-phone using finite-state transducers. In: Proc. IEEE Workshop on Speech Synthesis, Santa Monica, CA, USA (2002)
5. Daelemans, W., Bosch., A.: Tabtalk: Reusability in data-oriented grapheme-to-phoneme conversion. In: In Proceedings of Eurospeech (1993)
6. Bosch, A., Daelemans, W.: Data-oriented methods for grapheme-to-phoneme conversion. In: Proceedings of the sixth conference on European chapter of the Association for Computational Linguistics, Utrecht, The Netherlands (1993)
7. Daelemans, W., Bosch, A.: Language-independent data-oriented grapheme-to-phoneme conversion. In Progress in Speech Synthesis, J. P. van Santen, R. W. Sproat, J. P. Olive, and J. Hirschberg, Eds. Springer-Verlag (1997)
8. Han, E-H., Karypis, G.: Centroid-based document classification: analysis and experimental results. In: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 424-431 (2000)
9. Reichel, Uwe D. and Florian Schiel. Using morphology and phoneme history to improve grapheme-to-phoneme conversion. In: Proceedings of the InterSpeech, pp. 1937-1940 (2005)
10. Taylor, Paul. Hidden Markov Models for grapheme to phoneme conversion. In: Proceedings of the InterSpeech, pp. 1973-1976 (2005)
11. Yvon, Francois. Self-learning techniques for grapheme-to-phoneme conversion. In: Proceeding of the 2nd Onomastica Research Colloquium, London (1994)

Machine Translation

Long Distance Revisions in Drafting and Post-editing

Michael Carl[12], Martin Kay[1] Kristian T.H. Jensen[2]

[1] Stanford University, H-STAR and Department of Linguistics
[2] Copenhagen Business School, Languages & Computational Linguistics, Frederiksberg, Denmark

Abstract. This paper investigates properties of translation processes, as observed in the translation behaviour of student and professional translators. The translation process can be divided into a *gisting*, *drafting* and *post-editing* phase. We find that student translators have longer gisting phases whereas professional translators have longer post-editing phases. Long-distance revisions, which would typically be expected during post-editing, occur to the same extent during drafting as during post-editing. Further, both groups of translators seem to face the same translation problems. We suggest how those findings might be taken into account in the design of computer assisted translation tools.

1 Introduction

In contrast to the large number of publications on MT post-editing, little research has been carried out on how translators review and post-edit their own translations. Lörscher[10], one of the pioneers in translation process research, points out:

> Solving translation problems is often carried out as a series of steps. Generally, subjects do not immediately reach solutions which they consider to be optimal. ... subjects generally use (linguistically) simple strategies first, and only when they turn out to be unsuccessful do the subjects employ more complex strategies. This procedure of the subjects complies with the generative principle whereby complex translation strategies are ... derived from simpler structures. (p:430)

Revision and post-editing of drafted translation are thus in order and indicative of the complexity (or uncertainty) of a translation problem. Only few years ago, research on human translation processing was based on think-aloud protocols [4,9,10], however, recent technological developments have made it possible to directly analyse user activity data (UAD), notably eye movement data and keystroke data [5,3].

In a recent study, Malkiel [11] investigates the predicatability of "self-revisions" in English-Hebrew translations, based on manual analysis of

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 193-204

Received 21/11/09
Accepted 16/01/10
Final version 10/03/10

the revision keystrokes. In this paper, we use our triangulation technology [2,3] and discuss a method to automatically detect and analyse revision patterns.

Given the increasing interest in interactive Machine Translation, [8,13][3] and in the design of man-machine interfaces, we expect that insights derived from the study of human translation processing will provide valuable information for the designers of MT post-editing tools.

2 Gisting, Drafting and Post-editing

We base our research on a translation experiment [7] in which 12 professional and 12 student translators produced translations using the Translog [5] software.[4] Translog presents the source text (ST) in the upper part of the monitor, and the target text (TT) is typed in a window in the lower part of the monitor. When the start button is pressed, the ST is displayed and eye movement and keystroke data are registered. The task of the translator is then to type the translation in the lower window. After having completed the translation, the subject presses a stop button, and the translation, along with the translation process data, are stored in a log file.

Translators vary greatly with respect to how they produce translations. However, the process can be divided into three phases, which we refer to as *gisting*, in which the translator acquires a preliminary notion of the ST, *drafting* in which the actual translation is typed (drafted), and *post-editing* in which some or all of the drafted text is re-read, typos corrected and sentences rearranged or reformulated on the background of the translator's better understanding of the text by the time this stage is reached.

2.1 Translation Progression Graphs

The UAD can be represented in so-called translation progression graphs[12]. Figure 1 shows translation progression graphs for two students (S17 and S23) at the top and at the bottom respectively and a professional (P1) in the middle. The graphs plot activity data which was collected during the translation of a 160 words text from English into Danish.[5]

[3] Google has just made available a toolkit for human assisted translation with more than 50 languages.

[4] The software can be downloaded from `www.translog.dk`

[5] The English source text is shown in the Appendix.

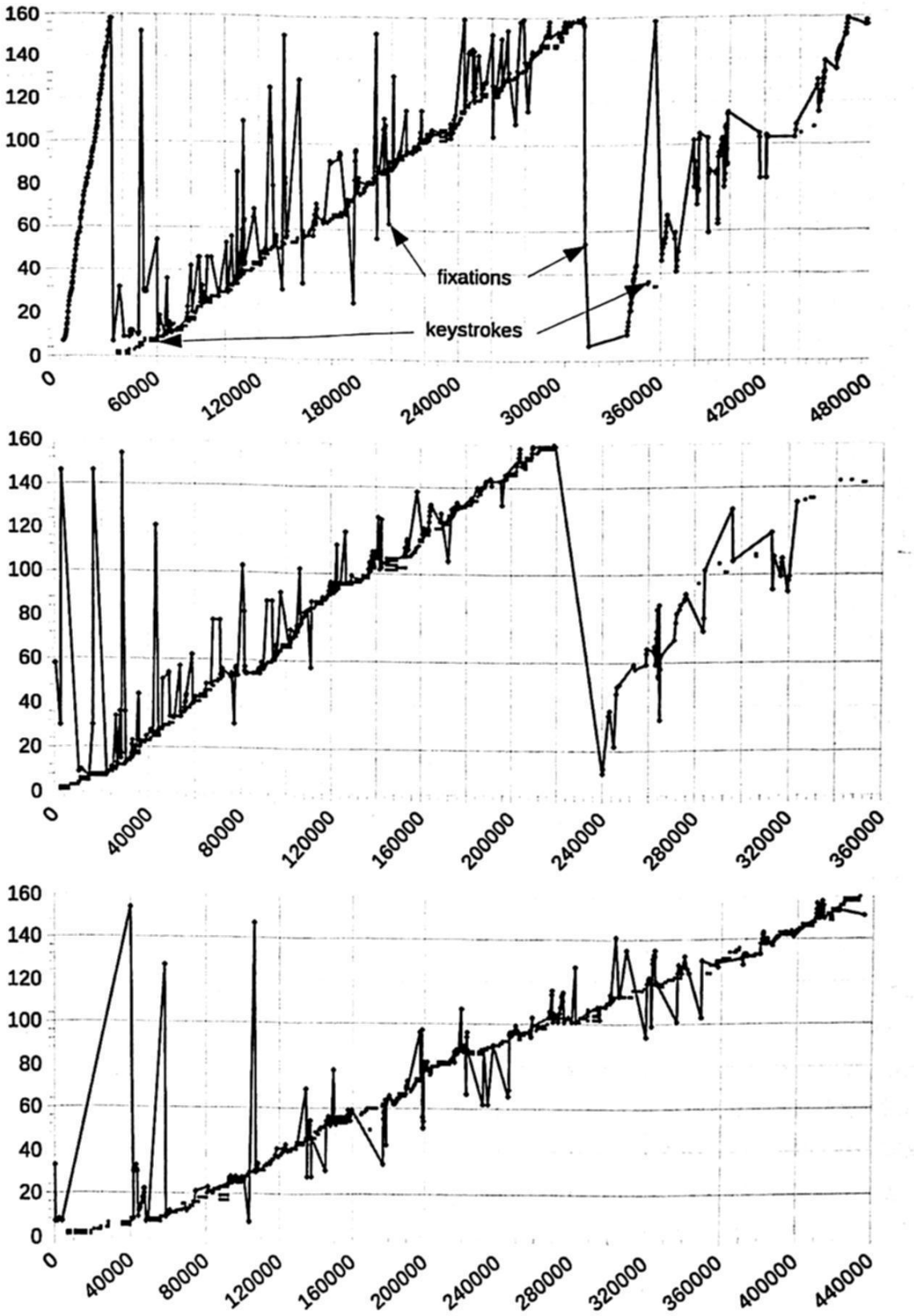

Fig. 1. Three translation progression graphs from top down subjects S17, P1 and S23, showing keystrokes and eye movements: S17 shows a clear distinction into gisting, drafting and post-editing. P1 has no gisting phase and spends almost 50% of the translation time on post-editing, while S23 only has a drafting phase.

The horizontal axis represents the translation time in milliseconds, and the vertical axis represents the source-language words from the beginning of the text (bottom), to the end (top). As described in Carl, 2009 [2], keystrokes that contribute to the TT, are mapped onto the ST words which they translate. All keystrokes that contribute to the translation of the ith source word are represented as single dots in the ith line from the bottom of the graph. The red (i.e. grey) line plots the gaze activities on the source text words. Single eye fixations are marked with a dot on the fixation line[6].

The progression graph of subject S17 (top graph in figure 1) shows a clear distinction between gisting, drafting and post-editing. Subject S17 spends almost 40 seconds getting acquainted with the text. The graph shows the progression of fixations nicely in which the ST is apparently read from beginning to end.

The drafting phase takes place between seconds 40 and 320. Eye movements can be observed where the translator moves back and forth between the ST and the TT. Some fixations are captured during this journey between the current ST position and the TT window (or to the keyboard) which are mapped on text positions remote from the current location of the corresponding translation.

The drafting phase is followed by a post-editing phase, from approx. second 320 until second 480. Translator S17 seems to re-read much of the ST during post-editing, but only few keystrokes occur, i.e. around seconds 360 and 440.

Translator P1, the second graph in figure 1, shows virtually no gisting phase. The first keystrokes can be observed less than 5 seconds after the ST appears on the screen. P1 also has a long post-editing phase of two minutes, from seconds 220 to 360. A number of revision keystrokes are visible, around seconds 300 and 340.

A third translation pattern for translator S23 is shown in final graph. No gisting and no post-editing take place, but some revision occurs at various places, e.g. around seconds 100 and 170. The time it takes to produce the translations is between 6 minutes (P1) and 8 minutes (S17).

2.2 Translation Expertise and Translation Phases

For students, there is a clear tendency towards longer gisting and shorter post-editing phases, whereas professional translators have shorter gisting

[6] Notice that only fixations on the source text are represented in the graph. Our software was not able to compute and map fixations on the emerging target text words.

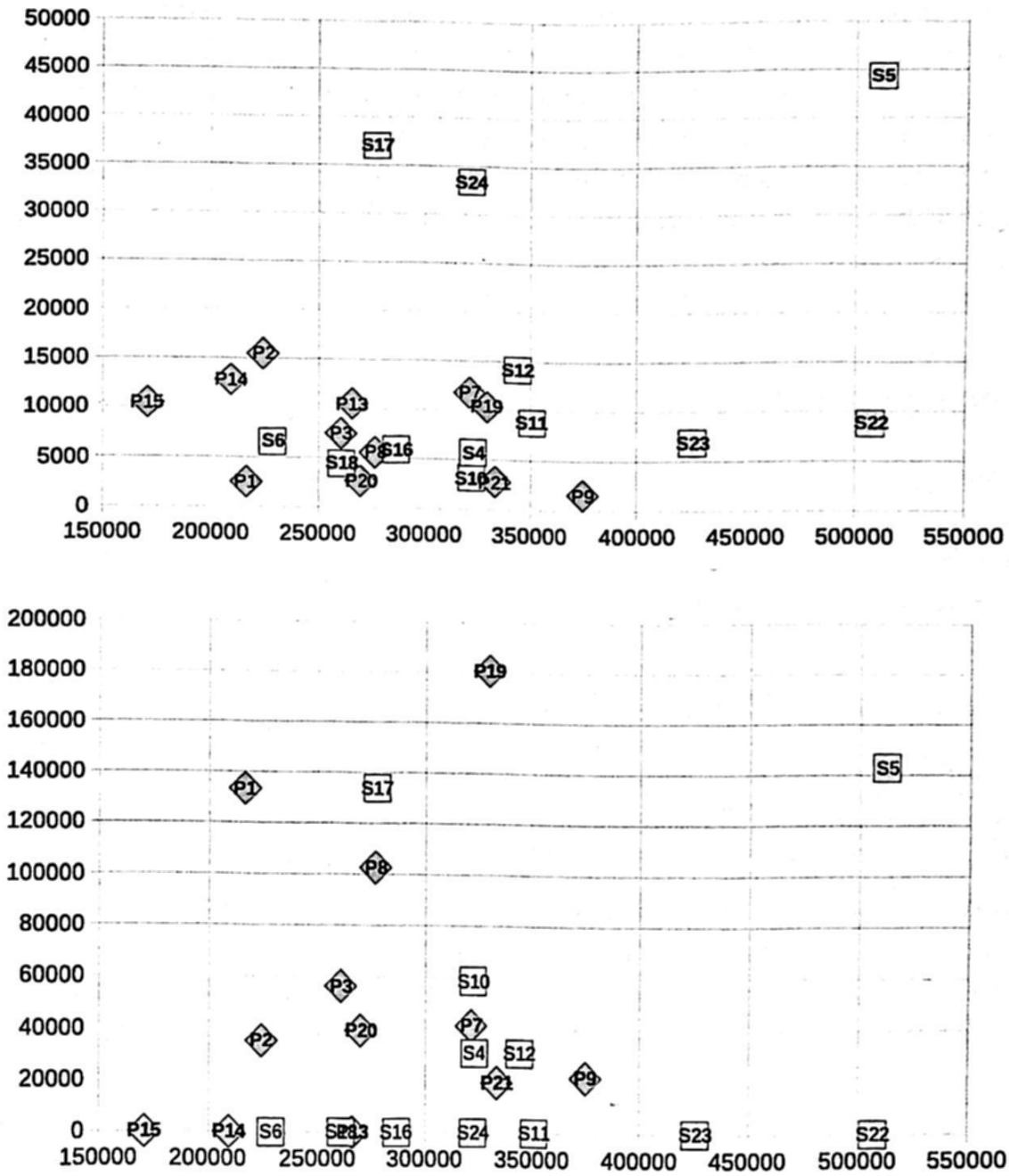

Fig. 2. Top: drafting time (horizontal) and gisting time (vertical). Rectangular symbols represent student translators, diamond shapes represent professionals. Students spend more time on gisting than professionals. Bottom: drafting time (horizontal) and post-editing time (vertical). Rectangular symbols represent students, diamond shapes represent professionals. On average, professionals spend more time post-editing than do students; many students completely skip post-editing.

and longer post-editing phases. Figure 2.1 (top) plots the relationship between drafting and gisting time, and the bottom graph in figure 2.1 shows the relation between drafting time and post-editing time. Almost all professional translators (9 out of 12) engage in some kind of post-editing, while 7 out of 12 student translators do not post-edit. The inverse observation can be made with respect to gisting: 3 students but no professional translator engage in gisting for more than 20 seconds. These results are only partially in line with Jakobsen, 2002 [6] who finds that professional translators invest more time than students in gisting and post-editing, but are faster at drafting the translation.

3 Long Distance Revisions

Changes in the target text translation may take place at any moment during drafting or post-editing: in the middle or at the end of a word or after or at the end of a sentence or paragraph. We distinguish between two types of revisions, short-distance revisions, and long-distance revisions.

3.1 Translation Phases and Long Distance Revisions

Long-distance revisions occur if two successive keystrokes are located 2 or more words apart from each other. For instance, a translator might first translate "nurse" into "sygeplejerske", but when she realizes that the 'nurse' is in fact masculin, she might correct all occurrences into "sygeplejer"[7]. To do so, the cursor must move to a previous words, and if the corrected word is two or more words apart from the last cursor action we will observe long-distance keystrokes. A long-distance revision is thus a sequence of two successive keystrokes, which are located in a different part of the target text translation. All other modifications of drafted text are short-distance revisions. Whereas short-distance revisions most likely are associated with typing errors, which the translator immediately corrects, it is plausible that long distance revisions are indicative of 'real' translation problems that the translator is struggling with.

One would expect that long-distance revisions are particularly abundant during post-editing; however, our data indicate that they occur with the same frequency although no separate post-editing phase takes place.[8] Figure 3 suggests that the post-editing time and the number of long-distance revisions are basically independent: long-distance revisions take

[7] In our classification below this would correspond to a IDWX pattern.

[8] An example of this is subject S23 in figure 1, above.

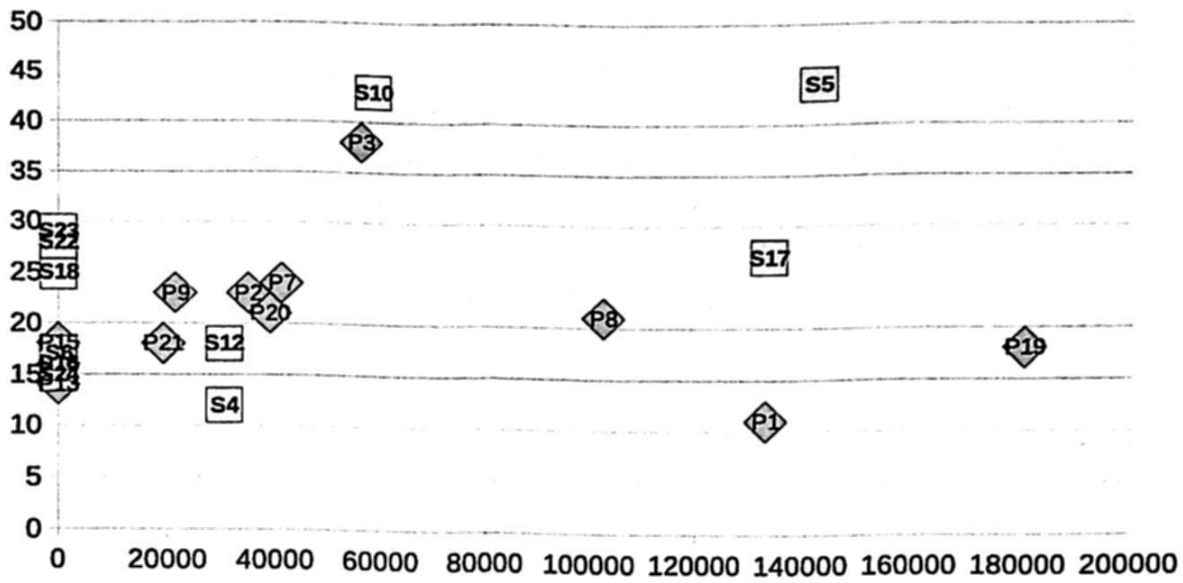

Fig. 3. Number of long distance revisions (vertical) and post-editing time (horizontal) shows the parameters to be unrelated. Long distance revisions occur equally frequently for students as for professionals, irrespectively of the length of the post-editing phase.

place in approximately equal number, whether or not there is a separate post-editing phase. Thus, more experienced, professional translators seem to prefer a modular mode of working, in whicn both types of editing are separated in two clearly different phases. Conversely, students are more likely to mix those two phases. Jakobsen [6] reports similar findings in his experiments, where students produce more revisions during drafting.

Figure 3 also shows that translators perform between 11 and 45 long distance revisions on the 160 word text. Students perform slightly more revisions, on average one revision every 6.5 words, while professionals revise once every 7.8 word. This figure approximately coincides with the one given in Malkiel [11] whose student translators "self-revise" every 8th word. In the next sections we will show that these revision are by no means equally distributed in the text.

3.2 Patterns in Long Distance Revisions

A related question is whether and to what extend translators face the same difficulties during translation. That is, we may be confident that translators share similar problems if long distance revisions cluster at particular text positions so that common patterns can be observed in the UAD.

Indeed, figure 4 shows that revisions of the 24 translators occur more frequently at certain positions in the texts. The graph shows four or five positions where many revision take place, i.e. around word positions 14, 50, 105, 120 and 151. The contexts of these passages are shown in bold in the Appendix. We briefly discuss some of the difficulties that these particular passages might present to a translatior.

A word-for-word translation of "imprisoned for life today" would not be idiomatic in Danish. In order to find an idiomatically acceptable rendering of the expression, the translator would have to reorder the constituents and make different lexical choices.

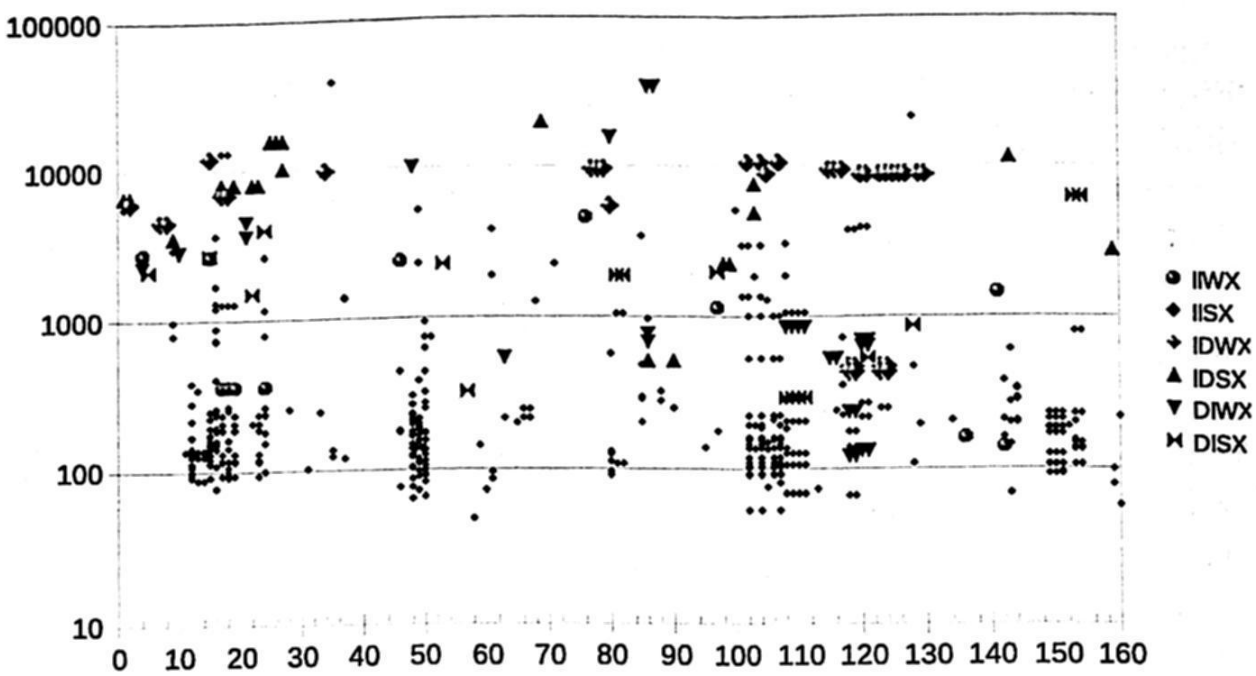

Fig. 4. Elapsed time (vertical) and positions of long distance revisions in the translation (horizontal): The horizontal axis enumerates the source-language words (0 to 160)and the dots in the graph represent different types of long distance revisions of their translations.

The translation of "counts of murder" into Danish may cause difficulty since the expression occurs infrequently in this context. The translator would have to test several Danish equivalent expressions in order find an acceptable one. The translation data shows more than 12 possible solutions for this passage.

The compound expression "hospital staff" has no exact equivalent in Danish. The translator would have to test several possible translation

alternatives before reaching a satisfying solution. This difficulty can also be measured by the fact that the data contain 20 different translations for "awareness of other hospital staff".

3.3 Classifying Long Distance Revisions

The keystrokes in our representations can be either text-inserting or text-deleting. That is, keystrokes for mere cursor movement are skipped and ignored in the graphs. Accordingly, in order to classify the long-distance revisions, we distinguish between insertion (I) and deletion (D) revisions. Since each of the two keystrokes in a revision can be an insertion or a deletion, we have four categories of pairs of revision keystrokes. In addition, we also distinguish the situation in which the second keystroke immediately follows a word separator (S) from the situation in which the second keystroke is in the middle of a word (W). Thus, in principle there could be eight types of long-distance revision.[9] The six most frequent combinations are shown in figure 4 and are briefly described below: [10]

1. IISX: two successive long-distance insertion keystrokes, the second immediately following a word separator, e.g. inserting an article.
2. IIWX: two successive long-distance insertion keystrokes, the second not immediately following a word separator, e.g inserting a suffix of a word.
3. IDSX: an insertion followed by a long-distance deletion keystroke occurring at the beginning of a word, e.g. deleting an article.
4. IDWX: an insertion followed by a long-distance deletion keystroke occurring in the middle of a word, e.g deleting a suffix.
5. DISX: a deletion followed by a long-distance insertion that occurs at the beginning of a dislocated word, e.g. inserting an article.
6. DIWX: a deletion followed by a long-distance insertion in the middle of a dislocated word, e.g inserting a suffix of a word.

Table 1 summarizes revision types for all 24 translations. It gives rise to the following observations: revisions usually start at the beginning of a word (461 occurrences) and less frequently in the middle (74 occurrences). ID revision patterns require much more time than DI or II revisions. That is, the time lapse between the end of an insertion and the beginning of a

[9] The long-distance between successive keystrokes is marked as X in the examples below.

[10] Unfortunately, our data show too few instances for 'DD' revisions to draw any conclusions.

deletion in another passage of the text is much higher than that between a deletion to a following insertion, or two successive insertions. On average, the pause between the insertion and the long-distance deletion is 7734ms and 8676ms respectively for the deletion to take place at the beginning and the middle of a word while it is only a fraction of this for the other types of revisions.

Type	IIWX	IISX	IDWX	IDSX	DIWX	DISX
Number of occurrences	12	423	35	20	27	18
Average time interval	755	169	8676	7734	689	1677

Table 1. Number of occurrences and time interval between the two keystrokes of for several types of long-distance revision pattern.

Presumably, the reason for the long ID revision is that a meaning hypothesis was realized and finished by the last insertion, and a new meaning hypothesis must mature before the deletion can take place. This would require much more anticipation and effort than a DI pattern, where the long-distance insertion is presumably only a consequence of the thought that lead to the deletion, or for the II patterns where the second insertion is a continuation of the first insterion.

4 Conclusion

Three phases can be distinguished in human translation: a *gisting phase*, a *drafting phase* and a *post-editing phase*. In our relatively short and simple text, gisting and post-editing seem to be optional: professional translators skip the gisting phase, tend to start immediately with drafting and have a longer post-editing phase. Novices, in contrast require a longer gisting phase, and often completely skip post-editing. In line with this, [7] finds that students "allocate considerably more time to each ST segment", our investigation indicates that this might be due to the longer gisting phase.

However, there seems to be an equal number of long-distance revisions for students and professionals. Hence, students revise parts of their translations when drafting, while professional translators work more structured and postpone revisions to a post-editing phase. Interestingly, irrespectively of when the revision is made, students and professionals revise the same parts of the translations, presumably because they face the same problems in the translation.

In order to figure out which of the phases in a translation process can be mechanized, computer assistance might be conceived to support the translator's structuring of the following task: gisting support tools could prepare the translator for difficulties of the ST, giving them e.g. a review of frequently used terms in their contexts, point to unusual collocations, etc., whereas translation memories or MT post-editing tools [8,13] might be a basis for drafting and post-editing support.

Special attention in the design of automated support during drafting and post-editing should receive the ID revision patterns, where translators spend much of their time here.

If certain translation and post-editing strategies turn out to be more successful than others, as in the case of our professional translators, then they should presumably be taken into account in the design of translation support tools. Under this assumption, a MT post-editing tool seems to be better grounded than a translation completion tool [1], which would mix drafting and post-editing phases, as we have observed in novice translators.

References

1. Sergio Barrachina, Oliver Bender, Francisco Casacuberta, Jorge Elsa, Civera Cubel, Shahram Khadivi, Antonio Lagarda, Hermann Ney, Jesús Tomás, Enrique Vidal, and Juan-Miguel Vilar. Statistical Approaches to Computer-Assisted Translation. *Computational Linguistics*, pages 3–28, 2009.
2. Michael Carl. Triangulating product and process data: quantifying alignment units with keystroke data. *Copenhagen Studies in Language*, 38:225–247, 2009.
3. Michael Carl and Arnt Lykke Jakobsen. Towards statistical modelling of translators' activity data. *International Journal of Speech Technology*, 12(4):124–146, 2010.
4. P. Gerloff. Second Language Learners Reports on the Interpretive Process: Talk-aloud Protocols of Translation. In *[?]*, pages 243–262, 1986.
5. Arnt Lykke Jakobsen. Logging target text production with Translog. In *[?]*, pages 9–20, 1999.
6. Arnt Lykke Jakobsen. Translation drafting by professional translators and by translation students. In *[?]*, pages 191–204, 2002.
7. Kristian T. H. Jensen. Distribution of attention between source text and target text during translation. In *IATIS*, 2009.
8. Philipp Koehn and Barry Haddow. Interactive Assistance to Human Translators using Statistical Machine Translation Methods. In *MT Summit*, 2009. `http://www.mt-archive.info/MTS-2009-TOC.htm`.
9. H. Krings. *Was in den Köpfen von Übersetzern vorgeht.* Gunter Narr, Tübingen, 1986.
10. W. Lörscher. Investigating the Translation Process . *Meta*, XXXVII(3):426–439, 1992.

11. Brenda Malkiel. From Ántona to My Ántona: tracking self-corrections with Translog. volume 37 of *Copenhagen Studies in Language*, pages 149–167. Copenhagen: Samfundslitteratur, 2008.
12. Daniel Perrin. Progression analysis (PA): investigating writing strategies at the workplace. *Pragmatics*, 35:907–921, 2003.
13. Marco Trombetti. Creating the World's Largest Translation Memory . In *MT Summit*, 2009. http://www.mt-archive.info/MTS-2009-TOC.htm.

Appendix: Source Test

Killer nurse receives four life sentences
Hospital Nurse Colin Norris was imprisoned for life today for the killing of four of his patients. 32 year old Norris from Glasgow killed the four women in 2002 by giving them large amounts of sleeping medicine. Yesterday, he was found guilty of four counts of murder following a long trial. He was given four life sentences, one for each of the killings. He will have to serve at least 30 years. Police officer Chris Gregg said that Norris had been acting strangely around the hospital. Only the awareness of other hospital staff put a stop to him and to the killings. The police have learned that the motive for the killings was that Norris disliked working with old people. All of his victims were old weak women with heart problems. All of them could be considered a burden to hospital staff.

Dependency-based Translation Equivalents for Factored Machine Translation

Irimia Elena, Alexandru Ceauşu

Reasearch Centre for Artificial Intelligence, Bucharest, Romania
{elena, aceausu}@racai.ro
www.racai.ro

Abstract. One of the major concerns of the machine translation practitioners is to create good translation models: correctly extracted translation equivalents and a reduced size of the translation table are the most important evaluation criteria. This paper presents a method for extracting translation examples using the dependency linkage of both the source and target sentence. To decompose the source/target sentence into fragments, we identified two types of dependency link-structures - super-links and chains - and used these structures to set the translation example borders. The option for the dependency-linked n-grams approach is based on the assumption that a decomposition of the sentence in coherent segments, with complete syntactical structure and which accounts for extra-phrasal syntactic dependency would guarantee "better" translation examples and would make a better use of the storage space. The performance of the dependency-based approach is measured with the BLEU-NIST score and in comparison with a baseline system.

Keywords. Lexical attraction model, statistical machine translation, translation model

1 Introduction

Corpus-based paradigm in machine translation has seen various approaches for the task of constructing reliable translation models,

- starting from the naïve "word-to-word" correspondences solution which was studied in the early works ([1], [2])
- continuing with the chunk-bounded n-grams ([3], [4], [5]) which were supposed to account for compounding nouns, collocations or idiomatic expressions,
- passing through the early approach of the bounded-length n-grams IBM statistical translation models and the following phrase-based statistical translation models ([6], [7], etc.),
- exploring the dependency-linked n-grams solutions which can offer the possibility of extracting long and sometimes non-successive examples and are able to catch the structural dependencies in a sentence (e.g., the accord between a verb and a noun phrase in the subject position), see [8],
- and ending with the double-sided option for the sentence granularity level, which can be appealing since the sentence boundaries are easy to identify but brings the

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 205-216

Received 28/11/09
Accepted 16/01/10
Final version 12/03/10

additional problem of fuzzy matching and complicated mechanisms of recombination.

Several studies were dedicated to the impact of using syntactical information in the phrase extraction process over the translation accuracy. Analyzing by comparison the constituency-based model and the dependency based model, [9] concluded that "using dependency annotation yields greater translation quality than constituency annotation for PB-SMT". But, as previous works ([10] and [11]) have noted, the new phrase models, created by incorporating linguistic knowledge, do not necessarily improve the translation accuracy by themselves, but in combination with the "old–fashioned" bounded-length phrase models.

The process of extracting syntactically motivated translation examples varies according to the different resources and tools available for specific research groups and specific language pairs. In a detailed report over the syntactically-motivated approaches in SMT, focused on the methods that use the dependency formalism, [12] distinguishes the situations when dependency parsers are used for both source and target languages from those in which only a parser for the source side is available. In the latter case, a direct projection technique is usually used to do an annotation transfer from the source to the target translation unit. This approach is motivated by the *direct correspondence assumption* (DCA, [13]), that states that dependency relations are preserved through direct projection. The projection is based on correspondences between the words in the parallel sentences, obtained through the lexical alignment (also called word alignment) process. Obviously, the quality of the projection is dependant of the lexical alignment quality. Furthermore, [13] notes that the target syntax structure obtained through direct projection is isomorphic to the source syntax structure, thus producing isomorphic translation models. This phenomenon is rarely corresponding to a real isomorphism between the two languages involved.

In the experiments we describe in this paper, we had the advantage of a probabilistic non-supervised dependency analyzer which depends on the text's language only through a small set of rules designed to filter the previously identified links. As both source and target dependency linking analysis is available, there is no need of direct projection in the translation examples extraction and the problem of the "compulsory isomorphism" is avoided.

2 Research Background

In previous experiments with an example-based approach on machine translation for the English-Romanian language pair, we developed a strategy for extracting translation examples using the information provided by a dependency-linker described in [14]. We then justified our opting for the dependency-linked n-grams approach based on the assumption in [15] that the EBMT potential should rely on exploiting text fragments shorter than the sentence and also on the intuition that a decomposition of the source sentence in "coherent segments", with complete syntactical structure, would be "the best covering" of that sentence.

The dependency-linker used is based on Yuret's Lexical Attraction Model (LAM, [16]), in who's vision the lexical attraction is a probabilistic measure of the combining affinity between two words in the same sentence. Applied to machine translation, the lexical attraction concept can serve as a mean of guaranteeing the translation examples usefulness. If two words are "lexically attracted" to one another in a sentence, the probability for them to combine in future sentences is significant. Therefore, two or more words from the source sentence that manifest lexical attraction together with their translations in the target language represent a better translation example than a bounded length n-gram.

The choice for the Yuret's LAM as the base for the dependency analyzer application was motivated by the lack of a dependency grammar for Romanian. The alternative was to perform syntactical analysis based on automatically inducted grammatical models. A basic request for the construction of this type of models is the existence of syntactically annotated corpora from which machine learning techniques could extract statistical information about the ways in which syntactical elements combine. As no syntactically annotated corpus for Romanian was available, the fact that Yuret's method could use LAM for finding dependency links in a not-annotated corpus made this algorithm a practical choice.

LexPar[14], the dependency links analyzer we used for the experiments described in this paper, is extending Yuret's algorithm by a set of syntactical rules specific to the processed languages (Romanian and English) that constraints the links' formation. It also contains a simple generalization mechanism for the link properties, which eliminates the initial algorithm inadaptability to unknown words. However, the LexPar algorithm does not guarantee a complete analysis, because the syntactic filter can contain rules that forbid the linking of two words in a case in which this link should be allowed. The rules were designed by the algorithm's author based on his observations of the increased ability of a certain rule to reject wrong links, with the risk of rejecting good links in few cases.

In our research group, significant efforts were involved in experimenting with statistical machine translation methodologies, focused on building accurate language resources (the larger the better) and on fine-tuning the statistical parameters. The aim was to demonstrate that, in this way, acceptable MT prototypes can be quickly developed and the claim was supported by the encouraging Bleu scores we obtained for the Romanian<->English translation system. The translation experiments employed the MOSES toolkit, an open source platform for development of statistical machine translation systems (see next section).

One of the goals of this paper was to analyze the impact of incorporating syntactic information in the translation model by means of a probabilistic dependency link analyzer. Although the non-supervised nature of the analyzer is affecting its recall, using this tool brings the advantage of having syntactic information available for translation without the need for training syntactically annotated corpora. We feed the Moses decoder with the new translation model and we compare the translation results with the results of the baseline system. In the remaining sections we will make a short survey of the resources and tools used in the SMT experiments (section 3), we will describe the dependency-motivated translation examples extraction process (section 4) and we will present the experiments and the results with the dependency-based translation model (section 5).

3 Factored Phrase-Based Statistical Machine Translation

The corpus. The Acquis Communautaire is the total body of European Union (EU) law applicable in the EU Member States. This collection of legislative text changes continuously and currently comprises texts written between the 1950s and 2008 in all the languages of EU Member States. A significant part of these parallel texts have been compiled by the Language Technology group of the European Commission's Joint Research Centre at Ispra into an aligned parallel corpus, called JRC-Acquis [17], publicly released in May 2006. Recently, the Romanian side of the JRC-Acquis corpus was extended up to a size comparable with the dimensions of other language-parts (19,211 documents)).

For the experiments described in this paper, we retained only 1-1 alignment pairs and restricted the selected pairs so that none of the sentences contained more than 80 words and that the length ratio between sentence-lengths in an aligned pair was less than 7. Finally, the Romanian-English parallel corpus we used contained about 600,000 translation units.

Romanian and English texts were processed based on the RACAI tools [18] integrated into the linguistic web-service platform available at http://nlp.racai.ro/webservices. After tokenization, tagging and lemmatization, this new information was added to the XML encoding of the parallel corpora. Figure 1 shows the representation of the Romanian segment encoding for the translation unit displayed in Figure 2. The tagsets used were compliant with the MULTEXT-East specifications Version3 [19] (for the details of the morpho-syntactic annotation, see http://nl.ijs.si/ME/V3/msd/).

```
<tu id="3936">
    ...
        <seg lang="ro">
            <s id="31985L0337.n.83.1">
                <w lemma="informaţie" ana="Ncfpry">Informaţiile</w>
                <w lemma="culege" ana="Vmp--pf">culese</w>
                <w lemma="conform" ana="Spsd">conform</w>
                <w lemma="art." ana="Yn">art.</w>
                <w lemma="5" ana="Mc">5</w>
                <c>,</c>
                <w lemma="6" ana="Mc">6</w>
                <w lemma="şi" ana="Crssp">şi</w>
                <w lemma="7" ana="Mc">7</w>
                <w lemma="trebui" ana="Vmip3s">trebuie</w>
                <w lemma="să" ana="Qs">să</w>
                <w lemma="fi" ana="Vasp3">fie</w>
                <w lemma="lua" ana="Vmp--pf">luate</w>
                <w lemma="în" ana="Spsa">în</w>
                <w lemma="considerare" ana="Ncfsrn">considerare</w>
                <w lemma="în cadrul" ana="Spcg">în cadrul</w>
                <w lemma="procedură" ana="Ncfsoy">procedurii</w>
                <w lemma="de" ana="Spsa">de</w>
                <w lemma="autorizare" ana="Ncfsrn">autorizare</w>
                <c>.</c>
```

```
            </s>
        </seg>
    ...
</tu>
```

Figure 1: Linguistically analysed sentence (Romanian) of a translation unit of the JRC-Acquis parallel corpus

Based on the monolingual data from the JRC-Acquis corpus we built language models for each language. For Romanian we used the TTL [20] and METT [21] tagging modelers. Both systems are able to perform tiered tagging [22], a morpho-syntactic disambiguation method that was specially designed to work with large (lexical) tagsets.

In order to build the translation models from the linguistically analyzed parallel corpora we used GIZA++ [23] and constructed unidirectional translation models (EN-RO, RO-EN) which were subsequently combined. After that step, the final translation tables were computed. The processing unit considered in each language was not the word form but the string formed by its lemma and the first two characters of the associated morpho-syntactic tag (e.g. for the wordform "informațiile" we took the item "informație/Nc"). We used for each language 20 iterations (5 for Model 1, 5 for HMM, 1 for THTo3, 4 for Model3, 1 for T2To4 and 4 for Model4). We included neither Model 5 nor Model 6, as we noticed a degradation of the perplexities of the alignment models on the evaluation data.

The MOSES toolkit [24] is a public domain environment, which was developed in the ongoing European project EUROMATRIX, and allows for rapid prototyping of Statistical Machine Translation systems. It assists the developer in constructing the language and translation models for the languages he/she is concerned with and by its advanced factored decoder and control system ensures the solving of the fundamental equation of the Statistical Machine Translation in a noisy-channel model:

$$Target^* = \text{argmax}_{Target} P(Source|Target)*P(Target) \quad (1)$$

The P(Target) is the statistical representation of the (target) language model. In our implementation, a language model is a collection of prior and conditional probabilities for unigrams, bigrams and trigrams seen in the training corpus. The conditional probabilities relate lemmas and morpho-syntactic descriptors (MSD), word-forms and lemmas, sequences of two or three MSDs. The P(Source|Target) is the statistical representation of the translation model and it consists of conditional probabilities for various attributes characterizing equivalences for the considered source and target languages (lemmas, MSDs, word forms, phrases, dependencies, etc). The functional *argmax* is called a decoder and it is a procedure able to find, in the huge search space P(Source|Target)*P(Target) corresponding to possible translations of a given Source text, the Target text that represent the optimal translation, i.e. the one which maximizes the compromise between the *faithfulness* of translation (P(Source|Target)) and the *fluency/grammaticality* of the translation (P(Target)). The standard implementation of a decoder is essentially an A* search algorithm. The current state-of-the-art decoder is the factored decoder implemented in the MOSES toolkit. As the name suggests, this decoder is capable of considering

multiple information sources (called factors) in implementing the *argmax* search. What is extremely useful is that the MOSES environment allows a developer to provide the MOSES decoder with language and translation models externally developed, offering means to ensure the conversion of the necessary data structures into the expected format and further improve them. Once the statistical models are in the prescribed format, the MT system developer may define his/her own factoring strategy. If the information is provided, the MOSES decoder can use various factors (attributes) of each of the lexical items (words or phrases): occurrence form, lemmatized form, associated part-of-speech or morpho-syntactic tag. Moreover, the system allows for integration of higher order information (shallow or even deep parsing information) in order to improve the output lexical items reordering. For further details on the MOSES Toolkit for Statistical Machine Translation and its tuning, the reader is directed to the EUROMATRIX project web-page http://www.euromatrix.net/ and to the download web-page http://www.statmt.org/moses/.

4 Extracting Translation Examples from Corpora (ExTRact)

In our approach, based on the availability of a dependency-linker for both the source and the target language, the task of extracting translation examples from a corpus contains two sub-problems: *dividing* the source and target sentences *into fragments* (according to the chosen approach) and *setting correspondences* between the fragments in the source sentence and their translations in the target sentence. The last problem is basically *fragment alignment* and we solved it through a heuristic based on lexical alignments produced by GIZA++.

The remaining problem was addressed using the information provided by LexPar, the dependency linker mentioned above. With a recall of 60,70% for English, LexPar was considered an appropriate starting point for the experiments (extending or correcting the set of rules incorporated as a filter in LexPar can improve it's recall).

Using MtKit, a tool specially designed for the visualization and correction of lexical alignments adapted to allow the graphical representation of the dependency links, we could study the dependency structures created by the identified links inside a sentence and we were able to observe some patterns in the links' behavior: they tend to group by nesting and to decompose the sentence by chaining. Of course, these patterns are direct consequences of the syntactical structures and rules involved in the studied languages, but the visual representation offered by MtKit simplified the task of formalization and heuristic modeling (see Fig. 1).

These properties suggest more possible decompositions for the same sentence, and implicitly the extraction of substrings of different length that satisfy the condition of lexical attraction between the component words.

Example 1: in Figure 1, from the word sequence "made in the national currency" can be extracted the subsequences: "national currency", "the national currency", „in the national currency", „made in the national currency". The irrelevant

sequences and those susceptible of generating errors (like "the national", "in the", "made in the national") are ignored.

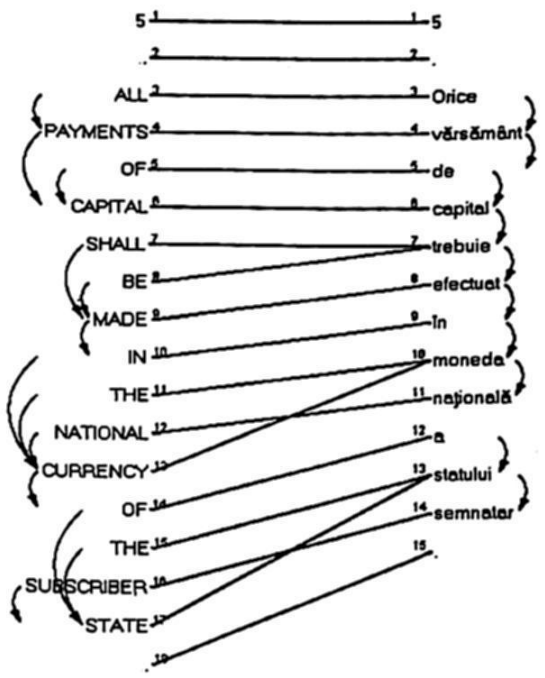

Fig. 2. MtKit visualisation of the alignments and links for an english-romanian translation unit. An arrow marks the existence of a dependence link between the two words it unites. The arrow direction is not relevant for the dependency link orientation.

The patterns observed above were formalized as *superlinks* (link structures composed of at least two simple links which nest, see Figure 3) and as *chains* (link structures composed of at least two simple links or superlinks which form a chain, see Figure 4).

Fig. 3. Superlink structures

Fig. 4. Chain structures

As input data, *ExTract* (the application that extracts translation examples from corpora) receives the processed corpus and a file containing the lexical alignments produced by GIZA++ [23]. We will describe the extracting procedure for a single

translation unit U in the corpus, containing *Ss* (a source sentence) and its trans-lation *Ts* (a target sentence). Starting from the first position in Ss (Ts respectively) we identify and extract every possible chaining of *links* and *superlinks*, with the condition that the number of chain loops is limited to 3. The limitation was introduced to avoid overloading the database. Subsequent experiments showed that increasing the limitation to 4 or 5 chains did not significantly improve the BLEU score of the translation system. Two list of candidate sentence fragment, from Ss and Ts, are extracted.

Every fragment in both sentences is projected through lexical alignment in a word string (note that this is not the direct syntactical structure projection discussed above) in the other language. A projected string of a candidate fragment in Ss is not necessarily part of the list of candidate sentence fragments Ts, and vice versa (LexPar is not able to identify all the dependency links in a sentence, the lexical alignments are also subject to errors). But if a fragment candidate from Ss projects to a fragment candidate from Ts, the pair has a better probability of representing a correct translation example. In this stage, the application extracts all the possible translation examples (*<source fragment candidate, projected word string>, <projected word string, target fragment candidate>)* but distinguish between them, associating a "trust" flag f="2" to the translation examples of the form *<source fragment candidate, target fragment candidate>*, and a flag f="1" to all the other. Thereby, it is possible to experiment with translation tables of different sizes and different quality levels.

5 Experiments and Results

Taking into account results from previous works ([12],[13]) that proved that dependency-based translation models give improved performance in combination with a phrase-based translation model, we decided to conduct our experiments in a mixed frame: we extracted from the dependency-based translation model only the translation examples longer than *2 source words <-> 2 target words,* creating a reduced dependency-based translation model and we combined it with the phrase-based translation model generated with the Moses toolkit.

Starting from the reduced D-based translation model, we can develop two different translation tables, based on the "trust" flags we introduced before:

- a *trustful D-based translation table* (if we keep only the examples with the flag f="2")
- a *relaxed D-based translation table* (if we accept all the examples, irrespective of the flags).

As we previously mentioned, the initial working corpus contained around 600,000 translation units. From this number, 600 were extracted for tuning and testing. The tuning of the factored translation decoder (the weights on the various factors) was based on the 200 development sentence pairs using MERT [25] method. The testing set contains 400 translation units.

The evaluation tool was the last version of the NIST official mteval script[1] which produces BLEU and NIST scores [26]. For the evaluation, we lowered the case in both reference and automatic translations. The results are synthesized in the following table, where you can notice that our assumption that the *trustful* table would produce better results than the *relaxed* one was contradicted by evidence. We thus learned that a wider range of multi-word examples is preferable to a restricted one, even if their correctness was not guaranteed by the syntactical analysis.

Table 1. Evaluation of the dependency translation table compared with the translation table generated with Moses (on unseen data)

Language pair	Moses translation table		Dependency translation table			
			Trustful table		Relaxed table	
	NIST score	BLEU score	NIST. score	BLEU score	NIST. score	BLEU score
English to Romanian	8.6671	*0.5300*	8.4998	0.5006	8.6900	*0.5334*
Romanian to English	10.7655	*0.6102*	10.3122	0.5812	10.3235	*0.6191*

As can be seen in the table, the translation accuracy obtained with the dependency-based translation table is very close to the one manifested by Moses, but still lesser. Therefore, we took a closer look at the translations and we noticed an important number of cases in which the dependency-based translation was more accurate in terms of human evaluation. Because of the space restriction, we will present here only a few of these cases and only for one direction of translation (English to Romanian). It can be noticed that the exact n-gram matching between the dependency-based translation and the reference is not as successful as the one between the Moses translation and the reference. But a flexible word matching, allowing for morphological variants and synonyms to be taken into account as legitimate correspondences, shows that the dependency-based translation is also very legitimate in terms of human translation evaluation.

English original:

the insurance is connected to a contract to provide assistance in the event of accident or breakdown involving a road vehicle;

whereas, in the light of experience gained, it is necessary to reconsider the consequences of the disposal of products from intervention on the markets of third countries other than those intended at the time of exportation;

the competent authorities of the member states shall afford each other administrative assistance in all supervisory procedures in connection with legal provisions and quality standards applicable to foodstuffs and in all proceedings for infringements of the law applicable to foodstuffs.

any administrative measure taken against an individual, leaving aside any consideration of general interest referred to above, on one of the grounds mentioned in article 1a, which is sufficiently severe in the light of the criteria referred to in section 4 of this joint position, may be regarded as persecution, in particular where it is intentional, systematic and lasting.

Romanian original:

[1] ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v12.pl

asigurarea priveşte un contract de acordare de asistenţă în caz de accident sau defecţiune a unui vehicul rutier;

întrucât, luând în considerare experienţa dobândită, este necesar să se reconsidere consecinţele desfacerii produselor de intervenţie asupra pieţelor din ţări terţe altele decât cele prevăzute în cazul exportului;

autorităţile competente din statele membre trebuie să îşi acorde reciproc asistenţă administrativă în toate procedurile de supraveghere prevăzute în dispoziţiile legale şi în normele de calitate aplicabile alimentelor, precum şi în toate procedurile privind încălcarea legislaţiei în domeniul produselor alimentare.

orice măsură administrativă luată împotriva unui individ, în afara considerentelor de interes general evocate mai sus, datorită unuia dintre motivele menţionate în art. 1a, care este suficient de severă potrivit criteriilor enunţate în secţiunea 4 din prezenta poziţie comună, poate fi considerată ca persecuţie, în special când aceasta prezintă un caracter intenţional, sistematic şi durabil.

Moses translation :

asigurarea este conectat la un contract să furnizeze asistenţă în caz de accident sau defecţiune a unui vehicul rutier ;

întrucât, ţinând seama de experienţa dobândită, este necesar să se reconsidere consecinţele comercializării produselor de intervenţie pe pieţele din ţările terţe, altele decât cele prevăzute în momentul exportului;

autorităţile competente ale statelor membre îşi acordă reciproc asistenţă administrativă în toate procedurile de supraveghere legate de dispoziţiile legale şi standardele de calitate se aplică produselor alimentare şi în toate procedurile pentru încălcarea legii aplicabile produselor alimentare

orice măsură administrativă luată împotriva unui individ, lăsând din circuitul agricol orice cauză de interes general menţionat anterior, pe unul din motivele menţionate în art. 1a, care este suficient de grave în lumina criteriilor menţionate la punctul 4 din prezenta poziţie comună, pot fi considerate ca persecuţie, în special atunci când s-a intenţionat, sistematic şi de durată.

Dependency-based translation :

asigurarea priveşte un contract de asistenţă în caz de accident sau defecţiune a unui vehicul rutier;

întrucât, în lumina experienţei acumulate, este necesar să se reconsidere consecinţele comercializării produselor de intervenţie pe pieţele ţărilor terţe altele decât cele prevăzute în cazul exportului;

autorităţile naţionale competente din statele membre acorde reciproc asistenţă administrativă în toate procedurile prevăzute în dispoziţiile financiare şi ale standardelor de calitate aplicabile produselor alimentare şi în toate procedurile privind încălcarea legii aplicabile produselor alimentare.

orice măsură administrativă luată împotriva unui individ, exclusiv, în afara considerentelor de interes general menţionat anterior, pentru unul din motivele menţionate la articolul 1a, care este suficient de grave ţinând seama de criteriile enunţate în secţiunea 4 din prezenta poziţie comună, poate fi considerată ca persecuţie, în cazul în care este intenţionat, sistematic şi durabile.

5 Conclusions

We described in this paper our method of extracting translation examples from corpora based on the links identified with a statistical non-supervised dependency-linker. Although the evaluation results did not overcome the performance of the

Moses translation model, the scores are promising and they can be improved by increasing LexPar's recall. We also intend to evaluate the results using metrics more sensitive to morphology variations and synonymy (e.g. METEOR, [27]).

Acknowledgements

The work reported here is funded by the STAR project, financed by the Ministry of Education, Research and Innovation under the grant no 742.

References

1. Gale, W. and K. Church, 1991. Identifying Word Correspondences in Parallel Texts. In Proceedings of the 4th DARPA Speech and Natural Language Workshop, Pacific Grove, CA., pp. 152-157.
2. Melamed, I.D. 1995. Automatic Evaluation and Uniform Filter Cascades for Inducing N-best translation lexicons. In proceedings of the Third Annual Workshop on Very Large Corpora, Cambridge, England, pp. 184-198.
3. Kupiec, J. 1993. *An Algorithm for Finding Noun Phrases Correspondences in Bilingual Corpora.* In 31st Annual Meeting of the Association for Computational Linguistics, Columbus, OH., pages 23-30.
4. Kumano, A. and H. Hirakawa. 1994. *Building an MT dictionary from parallel texts based on linguistic and statistical information.* In COLING-94: Proceedings of the 15th International Conference on Computational Linguistics, Kyoto, Japan, pages 76-81.
5. Smadja, F., K.R. McKeown and V. Hatzivassiloglou. 1996. *Translating Collocations for Bilingual Lexicons: A Statistical Approach.* Computational Linguistics 22(1):1-38.
6. Och, F.-J., Ch. Tillmann and H. Ney. 1999. *Improved Alignment Models for Statistical Machine Translation.* In Proc. of the Joint Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC 99), pages 20–28, College Park, MD, June.
7. Marcu D. and W. Wong. 2002. *A Phrased-Based, Joint Probability Model for Statistical Machine Translation.* In Proceedings Of the Conference on Empirical Methods in Natural Language Processing (EMNLP 02); pages 133-139, Philadelphia, PA, July.
8. Yamamoto, K. and Y. Matsumoto. 2003. *Extracting translation knowledge from parallel corpora.* In: Michael Carl & Andy Way (eds.) Recent advances in example-based machine translation (Dordrecht: Kluwer Academic Publishers, 2003); pages 365-395.
9. Hearne, M., S. Ozdowska, and J. Tinsley. 2008. *Comparing Constituency and Dependency Representations for SMT Phrase-Extraction.* In Proceedings of TALN '08, Avignon, France.
10. Groves D. & Way A. 2005. *Hybrid Example-Based SMT: the Best of Both Worlds?* In Proceedings of ACL 2005 Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond, p. 183–190, Ann Arbor, MI.
11. Tinsley J., Hearne M. & Way A. 2007. *Exploiting Parallel Treebanks to Improve Phrase-Based Statistical Machine Translation.* In Proceedings of The Sixth International Workshop on Treebanks and Linguistic Theories (TLT-07), Bergen, Norway.
12. Ambati, V. 2008. *Dependency Structure Trees in Syntax Based Machine Translation*, 11-734 Spring 2008, Survey Report, http://www.cs.cmu.edu/~vamshi/publications/DependencyMT_report.pdf

13. Hwa, R., Ph. Resnik, A. Weinberg, C. Cabezas and O. Kolak. 2005. *Bootstrapping parsers via syntactic projection across parallel texts*. Nat. Lang. Eng., 11(3):311–325, September.
14. Ion, R. 2007. *Metode de dezambiguizare automată. Aplicaţii pentru limbile engleză şi română*. Teză de doctorat. Academia Română. Bucureşti.
15. Cranias, L., H. Papageorgiou and S. Piperidis. 1994. *A Matching Technique in Example-Based Machine Translation*. In Proceedings of the 15th conference on Computational linguistics - Volume 1, Kyoto, Japan 100–104.
16. Yuret, D. 1998. *Discovery of linguistic relations using lexical atrraction*. PhD thesis, Department of Computer Science and Electrical Engineering, MIT
17. Steinberger R., Pouliquen, B., Widiger, A., Ignat, C., Erjavec, T., Tufiş, D. 2006. *The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages*. In Proceedings of the 5th LREC Conference, Genoa, Italy, 22-28 May, pp. 2142-2147
18. Tufiş, D., Ion, R., Ceauşu, A., Ştefănescu, D. (2008). *RACAI's Linguistic Web Services*. In Proceedings of the 6th Language Resources and Evaluation Conference - LREC 2008, Marrakech, Morocco. ELRA - European Language Ressources Association. ISBN 2-9517408-4-0.
19. Erjavec, T. 2004. *MULTEXT-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora*. In: Proc. of the Fourth Intl. Conf. on Language Resources and Evaluation, LREC'04, ELRA, Paris, pp. 1535 – 1538
20. Ion, R. 2007. *Word Sense Disambiguation Methods Applied to English and Romanian*, PhD thesis (in Romanian), Romanian Academy, Bucharest, 138 p.
21. Ceausu Al. 2006. *Maximum Entropy Tiered Tagging*. In Janneke Huitink & Sophia Katrenko (editors), Proceedings of the Eleventh ESSLLI Student Session, pp. 173-179
22. Tufiş, D. 1999. *Tiered Tagging and Combined Language Models Classifiers*. In Václav Matousek, Pavel Mautner, Jana Ocelíková, and Petr Sojka, editors, *Text, Speech and Dialogue (TSD 1999)*, Lecture Notes in Artificial Intelligence 1692, Springer Berlin / Heidelberg,. ISBN 978-3-540-66494-9, pp. 28-33.
23. Och, F. J., Ney H. 2000. *Improved Statistical Alignment Models*. In Proceedings of the 38th Conference of ACL, Hong Kong, pp. 440-447
24. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Wade, S., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E. 2007. *MOSES: Open Source Toolkit for Statistical Machine Translation*. Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session, Prague, Czech Republic.
25. Och, F. J. 2003. *Minimal Error Rate Training in Statistical Machine Translation*. In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, July 2003, pp. 160-167.
26. Banerjee S., Lavie A., *An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments,* Proceedings Of The ACL Workshop On Intrinsic And Extrinsic Evaluation Measures For Machine Translation And/Or Summarization, Pages 65-72, Ann Arbor, June 2005.

Information Retrieval and Text Clustering

Relation Learning from Persian Web: A Hybrid Approach

Hakimeh Fadaei[1], Mehrnoush Shamsfard[1]

[1]NLP Research Laboratory, Faculty of Electrical & Computer Engineering,
Shahid Beheshti University, Tehran, Iran
ha.fadaee@mail.sbu.ac.ir, m-shams@sbu.ac.ir

Abstract. In this paper a hybrid approach is presented for relation extraction from Persian web. This approach is a combination of statistical, pattern based, structure based and similarity based methods using linguistic heuristics to detect a part of faults. In addition to web, the developed system employs tagged corpora and WordNet as input resources in the relation learning procedure. The proposed methods extract both taxonomic and non-taxonomic, specific or unlabeled relations from semi-structured and unstructured documents.
In this system, a set of Persian patterns were manually extracted to be used in pattern base section. Similarity based approach which uses WordNet relations as a guide to extract Persian relations uses a WSD method to map Persian words to English synsets. This system which is one of the few ontology learning systems for Persian showed good results in performed tests. In spite of resource and tool shortage in Persian the results were comparable with methods proposed for English language.

Keywords: Relation learning, knowledge extraction, ontology learning, web, Wikipedia, similarity, Persian.

1 Introduction

Automatic extraction of semantic relations is a challenging task in the field of knowledge acquisition from text and is addressed by many researchers during recent years. As ontologies are widely used in many branches of science, building or enriching them is of great importance. Automatic methods for performing these tasks are so welcome since the process of building ontologies manually is very time consuming. There are no available ontologies for Persian and not much work is done on automatic extraction of ontological knowledge for this language.

In this paper we present a hybrid approach for extracting taxonomic and non-taxonomic relations from Persian resources. In the proposed system our focus is on using web as the learning resource although we use other resources to increase the effectiveness of our system too. The paper is organized as follows: In section 2 a brief review over related works is presented. The third section is dedicated to describing

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 219-230

Received 29/11/09
Accepted 16/01/10
Final version 10/03/10

our system and different methods used in it. Finally performed tests and their results are described in section 4.

2 Related Work

Automatic extraction of conceptual relations has attracted many attentions and some researchers work on proposing more efficient strategies in this field. During recent years different approaches were proposed to extract taxonomic and non taxonomic relations from different resources.

Pattern based, statistical, structure based and linguistic approaches are well-known approaches which extract relations from texts. Many systems (2, 3, 8) use combinations of these approaches to accumulate their advantages .

Pattern matching methods are widely used in extracting taxonomic and non-taxonomic relations. In this category, Hearst patterns [1] are among the most famous patterns defined for extracting taxonomic relations and has been used or adapted in many ontology learning systems 2, 3. Patterns may be defined manually [3] or extracted automatically 4. Some other systems (5, 14) use document structures to extract relations, these structures include tables, hyperlinks, html and xml tags and so on.

Some systems use statistical methods and rely on the distribution of words and their modifiers in text to extract relations 3, 6, 7 and 8. Linguistic structure of sentences is another source of information used in some systems 2, 8 and 9. Linguistic methods use morphological, syntactic and semantic analysis to extract relations. These methods need many linguistic tools like chunkers, taggers and parsers and are not easily used in languages such as Persian in which these tools are unavailable.

On the other hand, ontology learning systems use different resources to extract ontological knowledge, these resources include structured, semi structured on unstructured data. Raw or tagged texts are used by many systems such as 3, 5 and 6. Tagged corpora are proper resources for knowledge extraction as they are targeted for this task. These tags (POS, semantic or ...) helps systems to better detect relations but they are not available in all languages. During recent years many researchers are attracted to web as knowledge extraction resource.

The main reason that attracted the attentions of researchers to web documents for ontology learning is the huge amount of text in many languages that is available for everybody. Apart from availability and size there are some other features in web documents which make them suitable for the task of ontology learning and specially relation extraction.

Web documents are usually filled with structured and semi-structured data, tables and hyperlinks which can be used in the process of ontology learning, some systems like 5 and 14 use these structures to learn ontological knowledge. One of the other facilities provided by web is the existence of high performance and efficient search engines like Google which are used to search this large body of texts. Many systems like 2, 4, 8, 11 and 12 use search engines in the procedure of relation extraction. Wikipedia is another advantage of using web in ontology learning, since it has a collection of very informative short articles well suited for knowledge extraction. In systems like [5], 8 and 13 Wikipedia is used in relation extraction.

Beside the positive points mentioned above for web as a resource in the task of learning, web has some shortages as well. Web documents are mainly written by ordinary people with no NLP background and as they are not basically targeted for NLP applications they may need special processes in comparison with the corpora prepared by language experts.

3 The Proposed Hybrid Approach

Our proposed approach is a combination of statistical, pattern based, linguistic, structure based and similarity based methods. These methods may be used in a serial or parallel order. In serial (sequential) order the output of one is the input of the other while in the parallel approach each method extracts some relations and then we choose the best one by voting. They can also be used separately to extract different types of relations. In this approach we use web to a great extent to be able to use its advantages, but to cover the dark sides of using web as a resource, we also use other resources such as corpora, dictionaries and WordNet. In this section we will describe each method in more details.

3.1 Structure Based Approach

The structure based part of our system uses the Wikipedia pages' structures such as tables, bullets and hyperlinks to extract relations. In many Wikipedia documents we can find some information given via bullets. This information usually shows some taxonomic relations. Given a Persian word the system follows these steps to extract relations from bulleted text:

1. Extract the Wikipedia article of the given word.
2. Find the bulleted parts and extracts their items.
3. Refine the items extracted in the second step by omitting stop words and prepositional phrases, finding the heads of nominal groups and .....
4. Make new relations between the title of bulleted part and the results of the third step.

The translations of some of these relations are presented in table 1.

Table 1. Translation of some extracted relations from bullets

Isa(Versailles , historical place)
Isa(car accident, event)
Isa(Islam, religion)
Isa (hypertension, disease)
Isa(suicide, death)

Disambiguation pages in Wikipedia are also so helpful in extracting taxonomic relations. While searching a polysemous word in Wikipedia, if there are separate

articles for each meaning of the word, Wikipedia brings a disambiguation page as the search result. In this page some or all of the meanings of the word are presented, usually with a brief explanation, in front of them. These explanations could be either a phrase or just a word indicating the parent of the word.

While searching the word "tree" in Wikipedia, in disambiguation page we come across the following meanings of the word "tree":

- Tree is a woody plant
- Tree structure, a way of representing the hierarchical nature of a structure in a graphical form
- Tree (data structure), a widely used computer data structure that emulates a tree structure with a set of linked nodes
- Tree (graph theory), a connected graph without cycles

From the above explanations we can extract the relations:

- isa(tree, woody plant)
- isa(tree, way of representing)
- isa(tree, computer data structure)
- isa(tree, connected graph)

Table 2 contains some of the extracted relations from disambiguation pages. The extracted relations are between Persian terms but to be understandable for all readers we mention the English translation of extracted relations throughout this paper.

Table 2. Translation of some extracted relations from disambiguation pages

Isa (milk, dairy product)
Isa (lion, Felidae)
Isa (valve, device)
Isa (electric charge, concept)
Isa (Municipality, administrative division)
Isa (watch , device)

3.2 Similarity Based Approach

The similarity based part of our system uses Persian or English synonyms of a given word to find its related words and it is based on the similarities among the synonyms' contexts. The input of this part is a Persian word and as output, it returns a set of candidate related words to the given Persian word. The relation extraction resource could be WordNet, Persian corpus, Wikipedia or other resources, according to the application. The system finds the parts of the resource, which are related to each synonym, the intersection of which leads us to new relations. The process of finding the related parts of resource and finding the intersection is defined regarding the type of the resource and the task in hand. In the rest of this section we present three similarity based methods, using three different resources to extract taxonomic or non-taxonomic relations. In these methods (apart from the one using WordNet) the type of extracted relations are not defined and the system just extracts related words. The type (label) of these relations can be found by using pattern based method (see section 3.3)

or by doing linguistic analysis. In this section we will show the application of this approach on different learning resources.

3.2.1 WordNet

In this part system uses WordNet relations as a guide for extracting Persian relations. The whole idea is to find WordNet synsets with a meaning close to the Persian word's, and then to translate the relations of these synsets to Persian. The problem is how to find these WordNet synsets which is solved by similarity based method. Although in this section we talk about extracting taxonomic relations, this method can be well adapted for any other WordNet relations. To extract relations having WordNet as a resource, the system follows the following steps:

1. Find the English equivalents (translations) of the Persian word using a bilingual dictionary.
2. Find the related WordNet synsets for each English equivalent.
3. Select the synset(s) with greatest number of words in common with all English equivalents.
4. Find the hypernym synsets of the selected synset(s) in step 3.
5. Translate the words in hypernym synset(s) to Persian.
6. Make new relations between the given Persian word and each translation from step 5.

It is worth mentioning that zero or more translations may be found for each English word or phrase in step 5. In this method the system uses a Persian to English dictionary which gives English equivalents for Persian synsets. The structure of the dictionary is as follows:

Persian Word	Sense Number	Persian Synonyms	English Synonyms

As we didn't have any English to Persian dictionary with the same structure, we decided to use this dictionary reversely to translate English words to Persian ones. So since the English words are not annotated with sense numbers, the reverse dictionary's structure is as follows:

English Word	Persian Word	Persian Sense Number	Persian synonyms

The fact that we can't distinguish different senses of an English word causes some problems in translating hypernym synset(s) to Persian, the system doesn't know which translations are related to the word's sense indicated in WordNet synset. So we should have a mechanism to find the correct translation of the English words. To solve this problem and to increase the precision we used a voting strategy. In this strategy all the words in hypernym synset are searched in three resources to find their translations: bilingual dictionary, Wikipedia and Wiktionary 16 which is a wiki based dictionary. Then within the all retrieved Persian equivalents for all the synset words, we choose the equivalent(s) with greatest frequency. The translation of English words can be

directly found by bilingual dictionary and by Wiktionary, but to find the translation of words via Wikipedia the word is searched in English Wikipedia and we see if the retrieved article has link to the related Persian article. If such a link exists, the title of the Persian Wikipedia 15 article is the translation of the original English word.

To more clarify this method we follow what system dose for the first sense of the Persian word "آموزش" "Amuzesh". According to the bilingual dictionary the English equivalents of this word are: pedagogy, pedagogics, learning, instruction, tuition, study, teaching, schooling, educating and education. The related synsets of these words are shown in table 3.

Table 3. English synsets for english synonyms of the word "آموزش"

English word	Related synsets	Common number
pedagogy	(teaching method, pedagogics, pedagogy) || (teaching, instruction, pedagogy) || (education, instruction, teaching, pedagogy, didactics, educational activity)	2, 3, 4
pedagogics	(teaching method, pedagogics, pedagogy)	2
learnign	(learning, acquisition) || (eruditeness, erudition, learnedness, learning, scholarship, encyclopedism, encyclopaedism)	1, 2
Instruction	(direction, instruction) || (education, instruction, teaching, pedagogy, didactics, educational activity) || (teaching, instruction, pedagogy) || (instruction, command, statement, program line)	1, 4, 3, 1
Tuition	(tuition, tuition fee) || (tutelage, tuition, tutorship)	1, 1
Study	(survey, study) || (study, work) || (report, study, written report) || (study) || (study) || (discipline, subject, subject area, subject field, field, field of study, study, bailiwick, branch of knowledge) || (sketch, study) || (cogitation, study) || (study)	1, 1, 1, 1, 1, 1, 1, 1, 1
Teaching	(teaching, instruction, pedagogy) || (teaching, precept, commandment) || (education, instruction, teaching, pedagogy, didactics, educational activity)	3, 1, 4
Schooling	(schooling) || (school, schooling) || (schooling)	1, 1, 1
Educating	No relevant synsets found	
education	(education, instruction, teaching, pedagogy, didactics, educational activity) || (education) || (education) || (education) || (ducation, training, breeding) || (Department of Education, Education Department, Education)	4, 1, 1, 1, 1, 1

The system should select the synset(s) which has more words in common with all the English equivalents i.e. we find the intersection of each synset and the set of English equivalents and we choose the synset with largest intersection. As result we reach the most similar English synset to our Persian word. The number of words in the intersection of each sysnet (which is a sign for similarity and is called as "common number") in our example are indicated respectively in table 3 and we can see that the synset (education, instruction, teaching, pedagogy, didactics, educational activity) hast the larget intersection with 4 common words, so this synset is selected as our target synset in wordNet.

When the target English synset is found we start mapping its relations to Persian. As we mentioned before for now we choose Hypernymy relation to find taxonomic relations for the given Persian word. So we find the hypernym synset(s) of the selected synset which is: (activity). Now we translate all the words in Hypernym synset to Persian, and as result we have some hypernymy relations between the Persian word "آموزش" "Amuzesh" and the translated words. In our example the relation isa(آموزش, فعالیت) (which means isa(instruction, activity)) is extracted.

In some cases no synsets are selected in step 3 and that occurs when all the synsets only cover one word among the English equivalents. In these cases system follows another strategy. In this alternative strategy step 1 and 2 are the same as the previous one and the strategy continues with the following steps:

3. Find the hypernym(s) of all of the synstes retrieved in step2.
4. Select the hypernym synset(s) with most frequency among the hypernyms found in step 3.
5. Translate the words in selected synset(s) of step 4.
6. Make new relations between the given Persian word and each translation from step 5.

3.2.2 Wikipedia

Another resource used in extracting conceptual relations via similarity based method is Wikipedia. In the whole Wikipedia articles each important word which has an article in Wikipedia itself, is linked to its related article. These linked words especially the ones locating in the first section of the text are usually related to the title of the document. We use this fact to extract some taxonomic and non-taxonomic relations. This method can be also categorized under structure based approach. The following steps are followed by system to extract relations from Wikipedia by this method:

1. Find the Persian synonyms of the given word.
2. Find the related Wikipedia articles to the given word and all its synonyms.
3. Extract hyperlinked words of each Wikipedia article of step 3.
4. Find the common hyperlinked words in all extracted Wikipedia documents.
5. Make new relations between the given word and the words found in step 4.

The translations of some of the extracted relations by this method are shown in table 4.

Table 4. Translations of some relations extracted from wikipedia using similarity based approach

Input Word	Related word
Instruction	School
Child	Human
Life	Death
Life	Birth
Calculus	Math
Child	Son

3.2.3 Corpus

The last resource used in the similarity based part is corpus. In this system a general domain corpus named Peykareh 10 is used which is a collection gathered form Ettela'at and Hamshahri newspapers of the years 1999 and 2000, dissertations, books, magazines and weblogs. This method which can be classified as a statistical method contains the following steps:

1. Finding the Persian synonyms of the given word.
2. Finding the words which co-occur with the given word and its synonyms (separately) and their co-occurrence frequencies.
3. Selecting the words among the results of step2 with a frequency above a given threshold.
4. Finding the words of step 3 which co-occur with the given word and all its synonyms.
5. Making new relations between the words extracted in step 4 and the given word.

The threshold used in third step is to increase the precision of the extracted relations. The test results showed us that 8% of the total frequency of the word (the input word or any of its synonyms) is a proper threshold. Some of the relations extracted by this method are presented in table 5.

Table 5. Translations of some relations extracted from corpus using similarity based approach

Input Word	Related Word
Football	Team
Why	Reason
Scene	Movie
Politics	Government
Man	Woman
Actor	Movie
Success	Failure
Increase	Decrease
Death	Life

3.3 Pattern Based Approach

In this section we describe the pattern based part of our relation learning system. This system exploits pattern based approach to extract both taxonomic and non-taxonomic relations from Persian texts. To extract taxonomic relations we define a set of 36 patterns containing the adaptation of Hearst patterns for Persian and some other new patterns. We have also extracted some patterns for some well known non-taxonomic relations such as "Part of", "Has part", "Member of" and "synonymy". The translations of some of these patterns are shown in table 6 (TW stands for target word).

In this system pattern matching method is used in two modes. In the first mode the system is given a pair of related words and the target is to find the type of relation between them. These related pairs are obtained by using structured based or similarity based methods as it was described in section 3.2. In this case the two words are substituted in each pattern and are searched in corpus or web to find the occurrences of the patterns i.e. TW and X in above patterns are given. By following this method the system would be able to detect the type of relation if it is a taxonomic relation or a non-taxonomic one for which we have a template.

Table 6. Translation of some patterns for extracting relations

Pattern	Relation	Pattern	Relation
TW is X.	Hypernymy	TW is a part of X	Part of
TW is a X	Hypernymy	TW includes X	Has part
TW is considered as X	Hypernymy	TW means X	Definition
TW is known as X	Hypernymy	TW is defined as X	Definition
TW is called X	Hypernymy	TW1 or TW2 or ... are	Synonymy
TW is named as X	Hypernymy	TW has X	Has

In second mode system is given just one word for which it should find some relations. So in this case the X part of the patterns is known and we should find TW by matching patterns over text. As we can hardly find the occurrences of these patterns through the corpus and since large corpora are not available for Persian we decided to use Wikipedia in the process of relation extraction. In the first section of Wikipedia articles we can usually find some occurrences of our patterns. To start the pattern matching phase we extracted the 1000 most frequent Persian nouns and extracted the Wikipedia articles related to these words. For each word the related article is searched for the phrases matching any of the patterns. The translations of some of the extracted relations by this method are mentioned in table 7.

Table 7. Translation of some extracted relations

Isa (Iran, country)
Isa (newspaper, publication)
Isa (water, liquid)
Isa (man, human)
Isa (pen, tool)
Has part (personality, specificity)
Has part (Tehran, Tajrish)
Has (Greece, history)
Has (Iraq, source)
Synonym (thought, idea)

It should be mentioned that these patterns are used for simple phrases and while encountering complex phrases having different syntactic groups, the precision of the method decreases. To avoid this problem some text processing tools (e.g. chunker) are needed to find the constituents of sentences. As there is no efficient chunker for Persian, we did some post-processings to eliminate incorrectly extracted relations. This phase includes eliminating the stop words, applying some heuristics such as

matching the head of the first noun phrase in the sentence with the head of the extracted TW in copular sentences, eliminating prepositional phrases for taxonomic relations, replacing long phrases with their heads and so on.

4 Experimental Results

The proposed methods were tested separately and the results are presented in this section. As it was mentioned in section 3.3 the second mode of pattern based approach is tested over 1000 most frequent nouns of Persian. The extracted relations were mainly taxonomic and the number of non-taxonomic relations was much less. Since the given words were not domain specific and no reference ontology was available, human evaluation was used to evaluate our system. The results of pattern based section were evaluated by an ontology expert and a precision of 76% was obtained. Most of the error rate in this part is related to the lack of efficient linguistic tools like chunker for Persian. Although some refinement strategies were applied as was described in section 3.3, but still more process is needed to refine the extracted relation. Despite the unavailability of linguistic tools our system has a precision comparable with English pattern matching systems like 2 and 4.

Test results in structure based approach shows a precision of 55% in extracting relations from bullet structures and 74% in relation extraction via disambiguation pages. The problems mentioned above are also present in structure based methods but a great portions of them in handled by following the same rules described in 3.3. For this method and also for pattern based approach we had no efficient way to count the whole number of relations in the input resource to calculate the recall of our system.

Similarity based approach was tested regarding the used resource. Again we used human evaluation and according to our ontology expert the similarity based approach using WordNet has a precision of 73% and a recall of 54%. The similarity based method which uses Wikipedia has a precision of 76% according to human evaluation. This method has high precision but the number of extracted relations was low like in other proposed methods using Wikipedia and this fact has two major reasons: the first is that the Persian Wikipedia is sparse and there are many words for which no Wikipedia articles are created. Only about 35% of the selected words had correspondent articles in Wikipedia. The second reason is that the most frequent Persian common nouns which were used in test are mostly abstract nouns and Persian Wikipedia articles are usually about proper nouns or concrete nouns.

The final test was performed on the output relations of similarity based approach which uses corpus as input. This method was tested on 300 most frequent common nouns of Persian and the results were verified by three ontology experts. As it was mentioned in section 3.2.3 to increase the precision of extracted relations we considered a threshold and we accepted the co-occurrence relations with a frequency above the defined threshold. To find the proper threshold we performed an initial test that showed us this threshold should be between 5% and 10%. Then we tested our system with different threshold between these boundaries and asked our experts to mark the extracted relations as "Acceptable" or "Unacceptable" to appear in a domain independent ontology. As we had no means to calculate the recall we compared different results regarding their precision and number of correctly extracted relations.

The test results are shown in figure 1 in which horizontal axis shows precision and vertical axis indicates the number of correctly extracted relations.

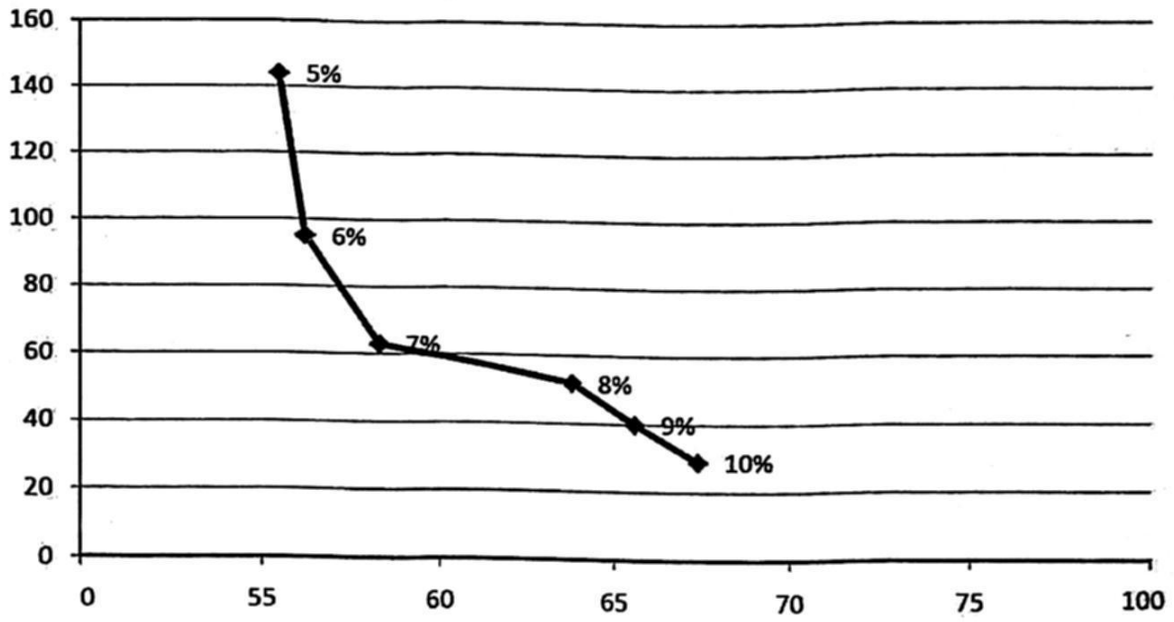

Fig. 1. Test results for different thresholds

As it can be seen in figure 1 the precision increases by raising the threshold while the number of correct relation decreases. To have a reasonable precision and not to miss many correct relations we decided to set 8% as our threshold.

5 Conclusion and Further Work

In this paper a hybrid approach was presented for extracting conceptual relations from Persian resources especially from web. This approach is a combination of pattern based, structure based, similarity based and statistical relations, enriched with linguistic heuristics. This system which is one of the few works done on ontology learning for Persian, uses different methods and resources to use their advantages and to cover the disadvantages of each method with others. The proposed approach is able to extract a noticeable number of relations and is used in the process of building Persian WordNet.

To increase the precision of extracted relations more linguistic heuristics could be applied. Extracting more patterns for taxonomic relations and covering more non-taxonomic relations could be considered as future work. Some more complex methods could be used to find the constituents of Persian sentences. In this way we can extract more relations, especially the non-taxonomic ones, from text and also the precision of pattern based method will increase. More advanced ways to find the types of relations via searching the web and linguistic analysis could be found.

References

1. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: 14th International Conference on Computational Linguistics, pp. 539--545, (1992).
2. Cimiano, P., Pivk, A., Schmidt-Thieme, L., Staab, S.: Learning Taxonomic Relations from Heterogeneous Sources of Evidenc. Ontology learning from text: Methods, Evaluation and Applications. IOS press (2005).
3. Shamsfard, M., Barforoush, A.: Learning Ontologies from Natural Language Texts. International journal of human- computer studies. 60, pp. 17--63 (2004).
4. D. Sanchez, A. Moreno, Discovering non-taxonomic relations from the Web. In: LNCS, vol. 4224, pp. 629--636. Springer, Heidelberg (2006).
5. Ruiz-Casado, M., Alfonseca, E., Okumura M., Castells, P.: Information Extraction and Semantic Annotation of Wikipedia. Ontology learning and Population: Bridging the Gap Between Text and Knowledge. IOS press (2008).
6. Reinberger, M., Spyns, P.: Unsupervised Text Mining for the Learning of DOGMA-Inspired Ontologies. Ontology learning from text: Methods, Evaluation and Applications. IOS press (2005).
7. Ryu, P., Choi, K.: An Information-Theoretic Approach to Taxonomy Extraction for Ontology Learning. Ontology learning from text: Methods, Evaluation and Applications. IOS press (2005).
8. Suchanek, F. M., Ifrim, G., Weikum, G.: Combining linguistic and statistical analysis to extract relations from web documents. In: 12th ACM SIGKDD international conference on Knowledge discovery and data mining, Philadelphia, PA, USA, (2006).
9. Ciaramita, M., Gangemi, A., Ratsch, E., Saric, J., Rojas, I.: Unsupervised Learning of Semantic Relations for Molecular Biology Ontologies. Ontology learning and Population: Bridging the Gap Between Text and Knowledge. IOS press (2008).
10. M. Bijankhan: Role of language corpora in writing grammar: introducing a computer software, Iranian Journal of Linguistics, No. 38 : 38-67, (2004).
11. Sanchez, D., Moreno, A.: Automatic Generation of Taxonomies from the WWW. In: 5th International Conference on Practical Aspects of Knowledge Management. Vol. 3336 of LNAI., pp. 208--219, (2004).
12. Cimiano, P. and Staab, S.: Learning by googling, ACM SIGKDD Explorations Newsletter, vol.6 n.2, p.24--33, (2004).
13. Herbelot, A., Copestake, A.: Acquiring Ontological Relationships from Wikipedia Using RMRS. In: Web Content Mining with Human Language Technologies workshop, USA, (2006).
14. Hazman, M., El-Beltagy, S.R. and Rafea, A.: Ontology learning from textual web documents. In: 6th International Conference on Informatics and Systems, NLP, pp.113--120, Giza, Egypt, (2008).
15. Persian Wikipedia, the free encyclopedia, http://fa.wikipedia.org
16. Persian Wiktionary, the free dictionary, http://fa.wiktionary.org

Towards a General Model of Answer Typing: Question Focus Identification

Razvan Bunescu and Yunfeng Huang

School of EECS
Ohio University
Athens, OH 45701
bunescu@ohio.edu, yh324906@ohio.edu

Abstract. We analyze the utility of *question focus* identification for answer typing models in question answering, and propose a comprehensive definition of question focus based on a relation of coreference with the answer. Equipped with the new definition, we annotate a dataset of 2000 questions with focus information, and design two initial approaches to question focus identification: one that uses expert rules, and one that is trained on the annotated dataset. Empirical evaluation of the two approaches shows that focus identification using the new definition can be done with high accuracy, holding the promise of more accurate answer typing models.

1 Introduction and Motivation

Open domain Question Answering (QA) is one of the most complex and challenging tasks in natural language processing. While building on ideas from Information Retrieval (IR), question answering is generally seen as a more difficult task due to constraints on both the input representation (natural language questions vs. keyword-based queries) and the form of the output (focused answers vs. entire documents). A common approach to the corresponding increased complexity has been to decompose the QA task into a pipeline of quasi-independent tasks, such as question analysis, document retrieval, and answer extraction. As part of question analysis, most QA systems determine the *answer type*, i.e. *the class of the object*, or *rhetorical type of discourse*, sought by the question [1]. For example, the question Q_1: *Who discovered electricity?* is looking for the name of a HUMAN entity, whereas Q_2: *What are liver enzymes?* asks for a DEFINITION type of discourse. The corresponding answer types will therefore be HUMAN, and DEFINITION respectively. Knowledge of the answer type associated with a given question can help during the answer extraction stage, when the system can use it to filter out a wide range of candidates. Moreover, the answer type may determine the strategy used for extracting the correct answer. The HUMAN answer type for question Q_1 means that the answer is simply the name of a person, possibly identified using a named entity recognizer. A DEFINITION question like Q_2, on the other hand, may involve strategies that identify paragraphs with definition structures focused on the question topic (*liver enzymes*), or more complex

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 231-242

Received 23/11/09
Accepted 16/01/10
Final version 11/03/10

strategies in which sentences on the question topic from multiple documents are automatically assembled into an answer paragraph that is given the rhetorical structure of a definition.

Most previous approaches to answer typing employ a predefined set of answer types, and use classifiers or manually crafted rules to assign answer types to questions. For example, [2] use a maximum entropy classifier to map each question into a predefined set of categories that contains all the MUC types, plus two additional categories: REASON for capturing WHY questions, and PHRASE as a catch-all category. Realizing the benefit of using more fine-grained categories, Li and Roth have proposed in [3] a more comprehensive set of answer types in the form of a two-level hierarchy, with a first level of 6 coarse classes that are further split into 50 fine classes on the second level. As pointed out by Pinchak and Lin in [4], using a predefined set of categories presents two major drawbacks:

1. There will always be questions whose answer types do not match any of the predefined categories, e.g. *What are the names of the tourist attractions in Reims* . Many question analysis systems employ a special catch-all category for these cases, which leads to a less effective treatment compared with the other categories.
2. The predetermined granularity of the categories leads to a trade-off between how well they match actual answer types and how easy it is to build taggers and classifiers for them. Thus, while it is relatively easy to tag names as instances of PEOPLE, this category is not a perfect fit for the question *Which former actor became president of the United States?*. Conversely, while an ACTOR answer type would be a better fit for this question, the corresponding tagging task during answer extraction will be more difficult.

As a solution to these problems, Pinchak and Lin introduced in [4] a probabilistic answer type model that directly computes the degree of fitness between a potential answer and the question context, effectively obviating the need for a predefined set of answer types. Their follow-up work in [5] presents an alternative approach to answer typing based on discriminative preference ranking. Like the probabilistic model in [4], the new flexible approach works without explicit answer types, and is shown to obtain improved results on a set of "focused" WHAT and WHICH questions.

Irrespective of whether they use an explicit set of answer types or not, many answer typing models emphasize the importance of one particular part of the question: the *question focus*, defined in [1] as "generally a compound noun phrase but sometimes a simple noun, that is the property or entity being sought by the question". According to [1], the nouns *city*, *population* and *color* are the focus nouns in the following questions:

Q_3 McCarren Airport is located in what *city*?
Q_4 What is the *population* of Japan?
Q_5 What *color* is yak milk?

The focus of a question is generally seen as determining its answer type. Singhal et al. [6], for example, use a lexicon that maps focus nouns to answer types.

They also give a syntactic rule for extracting the question focus from questions of the type *What X ...*, *What is the X ...*, and *Name the X ...*, according to which the focus is simply the syntactic head of the noun phrase *X*. Consequently, the nouns *company* and *city* constitute the focus nouns in the following two example questions taken from [6]:

Q_6 What *company* is the largest Japanese builder?
Q_7 What is the largest *city* in Germany?

The question focus is also important in approaches that do not use explicit answer types. The models of Pinchak et al. from [4, 5] compute how appropriate an arbitrary word is for answering a question by counting how many times the word appears in *question contexts*, where a question context is defined as a dependency tree path involving the *wh*-word. For a focused question such as *What city hosted the 1988 Winter Olympics?*, the authors observed that a question focus context such as *X is a city* is more important than the non-focus context *X host Olympics*.

Motivated by the observed importance of the question focus to answer typing models, in this paper we take a closer look at the associated problem of *question focus identification*. We first give an operational definition (Section 2), followed by a set of examples illustrating the various question categories that result from a question focus analysis (Section 3). We describe a rule-based system (Section 4.2) and a machine learning approach (Section 4.3) that automatically identify focus words in input questions, and compare them empirically on a dataset of 2,000 manually annotated questions (Section 5). The paper ends with a discussion of the results and ideas for future work.

2 What is the Question Focus?

To the best of our knowledge, all previous literature on answer typing assumes that a question has at most one instance of question focus. The examples given so far in questions Q_1 through Q_7 seem to validate this assumption. Accordingly, the question focus in many questions can be extracted using simple syntactic rules such as the noun phrase (NP) immediately following the WH-word (e.g. Q_3, Q_5, Q_6), or the predicative NP (e.g. Q_4, Q_7). However, the uniqueness assumption may be violated in the case of questions matching more than one extraction rule, such as Q_6 above, or Q_8 and Q_9 below:

Q_8 Which Vietnamese *terrorist* is now a UN *delegate* in Doonesbury?
Q_9 What famed London criminal *court* was once a feudal *castle* ?

One approach to enforcing uniqueness would be to rank the extraction rules and consider only the extraction from the top most matching rule. It is unclear though on which principles the rules would be ranked. Any relative ranking between the NP immediately following the WH-word and the predicative NP seems to be arbitrary, since questions Q_8 and Q_9 can be reformulated as questions Q_{10} and Q_{11} below:

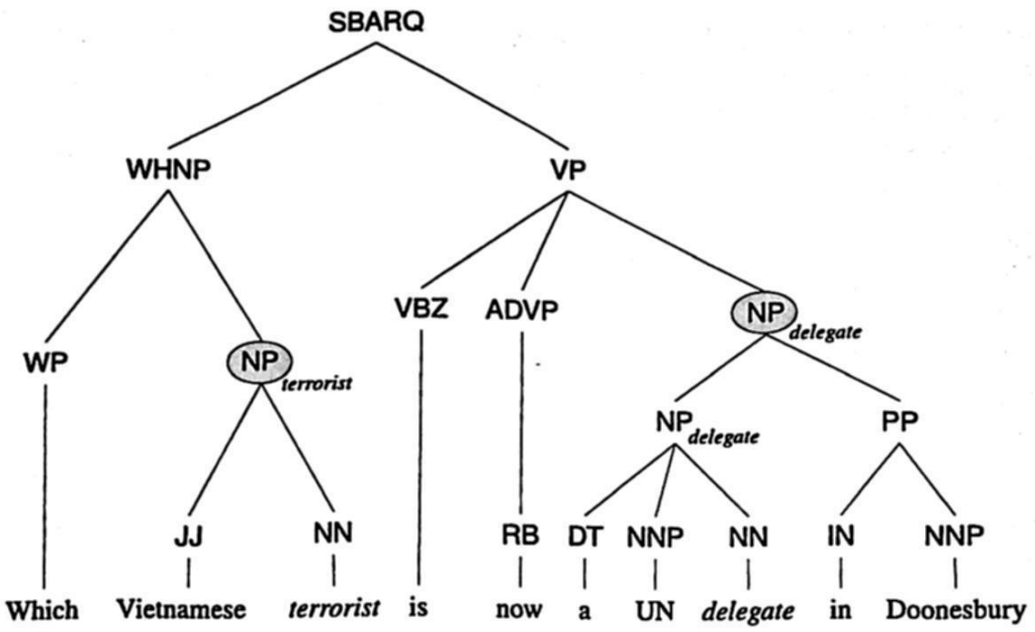

Fig. 1. Question focus example.

Q_{10} Which UN *delegate* in Doonesbury was once a Vietnamese *terrorist*?
Q_{11} What feudal *castle* is now a famed London criminal *court*?

In order to eschew these difficulties, we propose a definition of question focus that covers both the NP immediately following the WH-word and the predicative NP. For the definition to be as general as possible, it needs to take into account the fact that a question focus, as observed in [1], can denote *a property or entity* being sought by the answer. In question Q_6, for example, the focus word *company* specifies a property of the answer. In question Q_7, on the other hand, the noun phrase *the largest city in Germany* denotes the answer entity, while at the same time its head noun *city* specifies a property of the answer. Without exception, the noun phrases considered so far as potential instances of question focus have one thing in common: they can all be considered to *corefer* with the answer. It is this relation of coreference with the answer that allows us to give the following simple, yet comprehensive and operational definition for question focus:

Definition 1. *The question focus is the set of all maximal noun phrases in the question that corefer with the answer.* ■

Figure 1 shows the two noun phrases that are identified as question focus for question Q_8. The term *noun phrase* in the definition refers only to phrases marked as NP in the parse tree, and thus excludes *wh*-noun phrases (WHNP). A noun phrase is defined to be *maximal* if it is not contained in another NP with the same syntactic head.

In deriving the syntactic parse tree of a question, we use the same notation and bracketing criteria as the Penn Treebank [7], with one exception: if the WHNP contains more than a single *wh*-word, the rest of the phrase is abstracted

as an NP. This contrasts with the flat structure employed in the Penn Treebank, and helps in simplifying the question focus definition.

According to the definition, a question may have one or more instances of question focus. We believe that identifying more than one focus noun phrase can be advantageous in answer extraction, when all question focus instances may be used concurrently to filter out an even wider range of answer candidates. For example, when searching for noun phrases as potential answers to question Q_8, a question answering system may choose to enforce the constraint that the noun phrase refers to both a "Vietnamese *terrorist*" and a "UN *delegate* in Doonesbury". An answer typing system that employs an explicit set of answer types such as [3] may exploit the enriched question focus (e.g. {*terrorist*, *delegate*}) to improve the accuracy of mapping questions to answer types (e.g. HUMAN). Alternatively, the identification of multiple question focus instances may also benefit approaches that do not use a predefined set of answer categories. The answer typing methods of [4, 5], for example, may choose to give preference to dependency paths that start at any focus head in the question.

There are no constraints on the type of noun phrases that may be considered as question focus. Consequently, the focus may be a definite, indefinite, or bare noun phrase, as well as a proper name, or a pronoun, as illustrated in questions Q_8 (repeated below), Q_{12}, and Q_{13}:

Q_8 Which Vietnamese *terrorist* is now a UN *delegate* in Doonesbury?
Q_{12} What French *seaport* claims to be *The Home of Wines*?
Q_{13} Who was the first black *performer* to have *his* own network TV show?

The actual semantic constraints imposed on candidate answers by a question focus vary from alternate names (e.g. *The Home of Wines*), to category information (e.g. *seaport*), to pronouns (e.g. *his*). Even a simple personal pronoun such as *his*, when identified as a question focus, may trigger a useful elimination of candidate noun phrases that do not refer to entities that are both MALE and HUMAN.

3 Question Categories

When a question has at least one instance of question focus, as question Q_{14} below, the answer type can be determined from the focus. For questions such as Q_{15} that lack an explicit question focus, the answer type is implicit in the *wh*-word if it is one of *who*, *when*, *where*, or *why*. Question Q_{16} is an example where the answer type is both implicit in the *wh*-word, and explicit in the question focus, albeit at different levels of granularity. Finally, there are questions such as Q_{17} that do not contain any explicit question focus and where the *wh*-word does not convey any information about the answer type – except maybe as a negative implicature, e.g. since the question does not use the *wh*-word *who*, then it is unlikely that the answer is of type HUMAN.

Q_{14} What *country* do the Galapagos Islands belong to?

Q_{15} Who killed Gandhi?
Q_{16} Who was the *inventor* of silly putty?
Q_{17} What do bats eat?

The implicit answer type of *how* questions is MANNER (e.g. question Q_{18}), unless the *wh*-word is followed by an adjective or an adverb, as in questions Q_{19} and Q_{20} below. A full treatment of these *quantifiable how* questions is beyond the scope of this paper (Pinchak and Bergsma [8] have recently introduced an answer typing strategy specifically designed for such cases).

Q_{18} How does a rainbow form?
Q_{19} How successful is aromatherapy?
Q_{20} How long is the Coney Island boardwalk?

Using coreference to define a question focus implies an identity relationship between the question focus and the answer, which might not be as evident for questions Q_{21}, Q_{22}, or Q_{23} below. There is nevertheless an implicit identity relationship between the focus of these questions and their answers. Taking Q_{21} as an example, the answer is a text fragment with an appropriate rhetorical structure that describes some conceptual structure X that IS the "nature of learning".

Q_{21} What is the *nature* of learning?
Q_{22} What is the *history* of skateboarding?
Q_{23} What is the *definition* of a cascade?
Q_{24} What is a cascade?

Definition questions have a special status in this category, as their answer type can be expressed either explicitly through a question focus (Q_{23}), or just implicitly (Q_{24}).

4 Automatic Identification of Question Focus

Based on the definition from Section 2, a straightforward method for solving the task of question focus identification would contain the following two steps:

1. Run coreference resolution on the question sentence.
2. Select the coreference chain that is grounded in the answer.

In this section we present a more direct approach to question focus identification, in which every word of the question is classified as either belonging to the question focus, or not, leaving the coreference resolution based approach as subject of future work.

4.1 Question Focus Dataset

In order to evaluate our word tagging approaches to question focus identification, we selected the first 2000 questions from the answer type dataset of Li and Roth [3], and for each question we manually annotated the syntactic heads of all focus instances. Since, by definition, question focus identification is a constrained version of coreference resolution, we used the annotation guidelines of the MUC 7 coreference task [9]. Three statistics of the resulting dataset are as follows:

- 1138 questions have at least one instance of question focus.
- 121 questions have two or more instances of question focus.
- 29 questions have a pronoun as one instance of the question focus.

All 29 questions that have a pronoun as a question focus also contain a non-pronominal NP focus. This property, together with the relatively low occurrence of pronouns, determined us to design the initial extraction approaches to identify only non-pronominal instances of question focus.

4.2 A Rule Based Approach

In the first approach to question focus identification, we have manually created a set of extraction rules that correspond to common patterns of focused questions. The rules, together with a set of illustrative examples, are shown in Figure 2. Given that the syntactic information is not always correct, we have decided to associate each syntactic rule with an analogous rule in which some of the syntactic constraints are approximated with part-of-speech constraints. In most cases, this meant approximating the "head of an NP" with "the last word in a maximal sequence of words tagged with one of {JJX, NNX, CD}", where JJX refers to any adjective tag, and NNX refers to any noun tag. There are in total five syntactic rules R_1 to R_5, together with their part-of-speech analogues R'_1 to R'_5. A definite noun phrase is either a noun phrase starting with a definite determiner, a possessive construction, or a proper name. Whenever the focus is extracted as the head of a possessive construction in rules R_2 and R'_2, we modify the head extraction rules from [10] to output the "possessor" instead of the "possessed" noun (e.g. output *country* as head of *country's president*). We also use two small lexicons: one for BE verbs such as {*be, become, turn into*}, and one for NAME verbs such as {*name, nickname, call, dub, consider as, know as, refer to as*}.

4.3 A Machine Learning Approach

The rules R'_i in the rule based approach were construed to complement the syntactic rules R_i by approximating noun phrase constraints with constraints on sequences of part-of-speech tags. While they often lead to the extraction of focus words that otherwise would have been missed due to parsing errors, rules R'_i are generally expected to obtain lower precision and recall (see also Section 5). Ideally, each rule would be associated a weight, and a measure of confidence

1. If a question starts with the verb *Name*:
   R_1 = extract the head of the highest NP immediately after *Name*.
   R'_1 = extract the last word of the maximal sequence of {JJX, NNX, CD} immediately after *Name*.
   Q : *Name the scar-faced bounty hunter of The Old West.*

2. If a question starts or ends with *What/ Which* immediately followed by an NP:
   R_2 = extract the head of the highest NP immediately after the *wh*-word.
   R'_2 = extract the last word of the maximal sequence of {JJX, NNX, CD} immediately after the *wh*-word.
   Q : *What company is the largest Japanese builder?*
   Q : *The corpus callosum is in what part of the body?*

3. If a question starts with *What/ Which/ Who* immediately followed by a BE verb and does not end with a preposition or a past participle verb:
   R_3 = extract the head of the definite highest NP after the BE verb.
   R'_3 = extract the last word of the maximal definite sequence of {JJX, NNX, CD} after the BE verb.
   Q : *What company is the largest Japanese builder?*

4. If a question starts with *What/ Which/ Who*, optionally followed by a non-possessive NP, followed by a NAME verb in passive voice:
   R_4 = extract the head of the highest NP after the NAME verb.
   R'_4 = extract the last word of the maximal sequence of {DT, JJX, NNX, POS, CD} after the NAME verb.
   Q : *What city is sometimes called Gotham?*

5. If a question starts with *What/ Which/ Who*, followed by an interrogative pattern of a NAME verb:
   R_5 = extract the head of the highest NP after the NAME verb.
   R'_5 = extract the last word of the maximal sequence of [DT, JJX, NNX, POS, CD] after the NAME verb.
   Q : *What author did photographer Yousuf Karsh call the shiest man I ever met?*

Fig. 2. Focus Identification Rules and Example Questions.

would be computed for each candidate focus word based on the weights of the rules used to extract it. Such a setting can be obtained by modeling the focus identification task as a binary classification problem in which question words are classified as either part of the question focus or not. Each rule from the rule based approach would give rise to a binary feature whose value would be 1 only for the word positions matched by the rule. The fact that one word may be identified as focus by multiple rules will not be problematic for discriminative learning methods such as Support Vector Machines (SVMs) [11] or Maximum Entropy [12], which can deal efficiently with thousands of overlapping features. For our machine learning approach to focus identification we chose to use SVMs,

Question-level features:

1. The question starts with a preposition followed by a *wh*-word.
   Q : *In what U.S. state was the first woman governor elected?*
2. The question starts with *Name*.
   Q : *Name the scar-faced bounty hunter of The Old West.*
3. The question starts with a *wh*-word followed by a BE verb.
4. The question starts with *What/ Which* followed by a BE verb and a bare NP.
   Q : *What are liver enzymes?*
5. The question starts with *What/ Which* followed by a BE verb and ends with a past participle verb, optionally followed by a preposition.
   Q : *What is a female rabbit called?*
6. The question starts with a *wh*-word in an empty WHNP.
   Q : *What are liver enzymes?*
7. The question starts with a *wh*-word followed by an NP.
   Q : *What company is the largest Japanese builder?*
8. The question ends with a preposition.
   Q : *What are Cushman and Wakefield known for?*
9. The first verb after the *wh*-word is not a BE verb.
   Q : *Who killed Gandhi?*
10. The questions starts with ⟨*wh*-word⟩: create a feature for each possible *wh*-word.

Word-level features:

1. Create a feature for each of the rules $R_1, R'_1 \ldots R_5, R'_5$.
2. The head of the highest bare NP after the WHNP.
   Q : *What are liver enzymes?*
3. The head of the highest definite NP after the WHNP.
   Q : *What company is the largest Japanese builder?*
4. The head of the highest indefinite NP after the WHNP.
   Q : *What is a cascade?*
5. The head of the highest NP nearest the *wh*-word.
6. The last word of the first maximal sequence of {JJX, NNX, CD} nearest the *wh*-word.
   Q : *What is considered the costliest disaster the insurance industry has ever faced?*
7. The part-of-speech ⟨tag⟩ of the word: create a feature for each possible tag.

Fig. 3. Question-level and Word-level Features .

motivated by their capability to automatically induce implicit features as conjunctions of original features when run with polynomial or Gaussian kernels [13]. The explicit features employed in the SVM model are shown in Figure 3. Apart from the rules R_i and their analogues R'_i from Figure 2, the feature set also contains more atomic features designed to capture simple constraints used in the rules. Splitting rules into their more atomic features means that an isolated error in one part of the parse tree would only affect some of the features, while the rest of the features will still be relevant. Thus, while the rule as a whole may lead to an incorrect extraction, some of its atomic features may still be used

to reach a correct decision. Furthermore, if the SVM model is coupled with a polynomial kernel, then more complex parts of the original rules, when seen as conjunctions of elementary features, would be considered as implicit features.

5 Experimental Evaluation

We empirically evaluated the rule based approach and the learning approach on the task of question focus identification using the 2000 manually labeled questions. For the rule based approach, we evaluated 3 rule sets: the set of rules R_1 to R_5, their approximations R'_1 to R'_5, and all the rules from R_1, R'_1 to R_5, R'_5 in a combined set. For the learning approach, we performed 10-fold cross-validation by splitting the dataset into 10 equally sized folds, training on 9 folds and testing on the remaining 1 fold for 10 iterations. To compute the accuracy and F_1 measure, we pooled the results across all 10 folds. We used SVMLIGHT[1] with its default parameters and a quadratic kernel. Table 1 shows the precision, recall, F_1 measure, and accuracy for all four systems. The precision, recall and F_1 measures correspond to the task of focus word extraction and are therefore computed at word level. Accuracy is computed at question level by considering a question correctly classified if and only if the set of focus words found by the system is exactly the same as the set of focus words in the annotation.

Table 1. Experimental Results.

	Rule Based			SVM
Measure	R_1 to R_5	R'_1 to R'_5	Combined	Quadratic
Precision	93.3%	92.5%	88.1%	**95.2%**
Recall	89.2%	71.6%	90.1%	**91.3%**
F_1	91.2%	80.7%	89.1%	**93.2%**
Accuracy	91.1%	81.8%	88.3%	**93.5%**

As expected, adding the approximation rules to the original rules in the combined system helps by increasing the recall, but hurts the precision significantly. Overall, the learning approach obtains the best performance across all measures, proving that is can exploit useful combinations of overlapping rules and features. Figure 4 shows graphically the precision vs. recall results for the four systems. The curve for the SVM approach was obtained by varying a threshold on the extraction confidence, which was defined to be equal with the distance to the classification hyperplane, as computed by the SVMLight package.

A significant number of errors made by the learning approach are caused by parsing or part-of-speech tagging errors. There are also examples that are misclassified due to the fact that syntactic information alone is sometimes insufficient. For example, questions such as *What is a film starring Jude Law?* have

[1] URL: http://svmlight.joachims.org

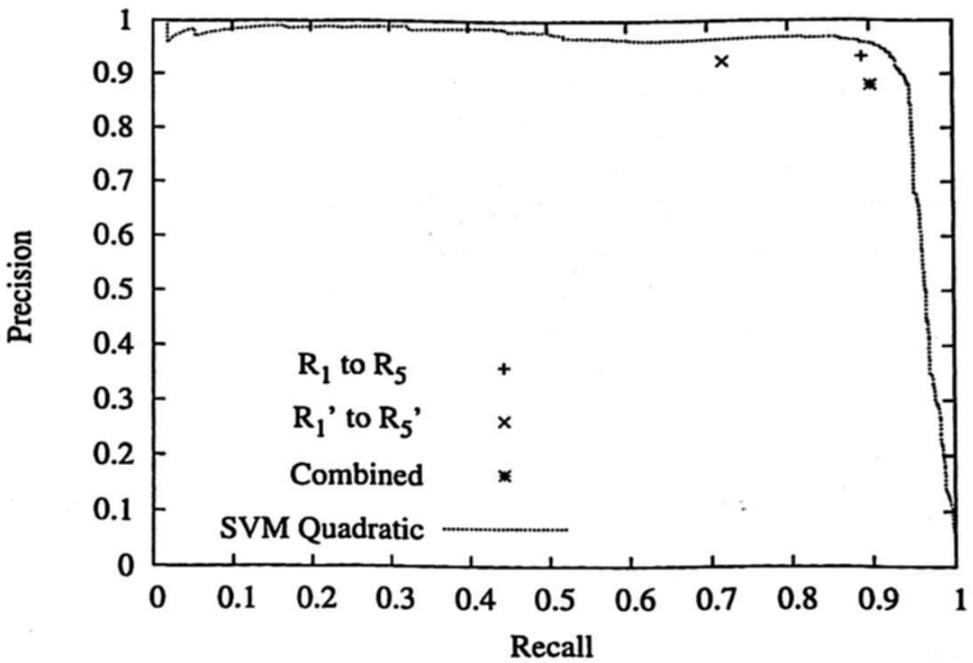

Fig. 4. Precision vs. Recall graphs.

the structure of an implicit definition question, yet they do contain an explicit focus word. The opposite is also possible: for the implicit definition question *What is the "7-minute cigarette"?* the system identifies *cigarette* as focus word. Semantic features that discriminate between proper names and titles may also eliminate some of the errors. The system learns, for example, that words tagged as NNP are unlikely to be focus words, which is why it fails to extract the focus word for the question *Who was President of Afghanistan in 1994?*.

Recently, Mikhailian et al. [14] have proposed a Maximum Entropy approach for identifying the question focus (the *asking point* in their terminology). However, they use the traditional, less comprehensive definition of question focus, whereby a question can have at most one focus noun phrase. In order to empirically compare our learning approach with theirs, we created a second version of the dataset in which the questions were annotated with at most one NP focus. Using exact matching between system output and annotations, our SVM based approach obtains a question level accuracy of 93.7%, which compares favorably with their reported accuracy of 88.8%.

6 Future Work

We plan to augment the SVM model with semantic features, some of them identified in Section 5, in order to further increase the accuracy. We also intend to implement the alternative method mentioned at the beginning of Section 4, in which the identification of question focus is done by classifying the coreference chains extracted from the question as referring to the answer or not.

7 Conclusions

We proposed a comprehensive definition of question focus based on coreference with the answer that eliminates inconsistencies from previous definitions. We designed both a rule based approach and a machine learning approach to question focus identification, and evaluated them on a dataset of 2000 questions manually annotated with focus information. Empirical evaluation of the two approaches shows that focus identification using the new definition can be done with high accuracy, offering the promise of more accurate answer typing models.

References

1. Prager, J.M.: Open-domain question-answering. Foundations and Trends in Information Retrieval 1 (2006) 91–231
2. Ittycheriah, A., Franz, M., Zhu, W.J., Ratnaparkhi, A., Mammone, R.J.: IBM's statistical question answering system. In: Proceedings of the Ninth Text REtrieval Conference, National Institute of Standards and Technology (NIST) (2000)
3. Li, X., Roth, D.: Learning question classifiers. In: Proceedings of the 19th International Conference on Computational linguistics, Taipei, Taiwan, Association for Computational Linguistics (2002) 1–7
4. Pinchak, C., Lin, D.: A probabilistic answer type model. In: Proceedings of the 11st Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, The Association for Computer Linguistics (2006)
5. Pinchak, C., Lin, D., Rafiei, D.: Flexible answer typing with discriminative preference ranking. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, Athens, Greece (2009) 666–674
6. Steve, A.S., Abney, S., Bacchiani, M., Collins, M., Hindle, D., Pereira, O.: Att at trec-8. In: In Proceedings of the Eighth Text REtrieval Conference (TREC-8. (2000) 317–330
7. Bies, A., Ferguson, M., Katz, K., MacIntyre, R.: Bracketing Guidelines for Treebank II Style Penn Treebank Project. University of Pennsylvania. (1995)
8. Pinchak, C., Bergsma, S.: Automatic answer typing for how-questions. In: Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics, Rochester, New York, Association for Computational Linguistics (2007) 516–523
9. DARPA, ed.: Proceedings of the Seventh Message Understanding Evaluation and Conference (MUC-98), Fairfax, VA, Morgan Kaufmann (1998)
10. Collins, M.: Head-driven Statistical Models for Natural Language Parsing. PhD thesis, University of Pennsylvania (1999)
11. Vapnik, V.N.: Statistical Learning Theory. John Wiley & Sons (1998)
12. Berger, A.L., Della Pietra, S.A., Della Pietra, V.J.: A maximum entropy approach to natural language processing. Computational Linguistics **22** (1996) 39–71
13. Schölkopf, B., Smola, A.J.: Learning with kernels - support vector machines, regularization, optimization and beyond. MIT Press, Cambridge, MA (2002)
14. Mikhailian, A., Dalmas, T., Pinchuk, R.: Learning foci for question answering over topic maps. In: Proceedings of the ACL-IJCNLP 2009 Conference, Suntec, Singapore, Association for Computational Linguistics (2009) 325–328

Traditional Rarámuri Songs used by a Recommender System to a Web Radio

Alberto Ochoa-Zezzatti[1] , Julio Ponce[2], Arturo Hernández[3], Sandra Bustillos[1], Francisco Ornelas[2] & Consuelo Pequeño[1]

[1]Instituto de Ciencias Sociales y Administración, Universidad Autónoma de Ciudad Juárez; México.
[2]Aguascalientes University; Aguascalientes, México.
[3]CIMAT; Guanajuato, México.
alberto.ochoa@uacj.mx, jcponce@correo.uaa.mx, artha@cimat.mx, fjornel@correo.uaa.mx

Abstract. In this research is described an Intelligent Web Radio associated to a Recommender System which uses different songs in a database related with a kind of Traditional Music (Rarámuri songs) which employs the Dublin Core metadata standard for the documents description, the XML standard for describing user profile, which is based on the user's profile, and on service and data providers to generate musical recommendations to a Web Radio. The main contribution of the work is to provide a recommendation mechanism based on this Recommender System reducing the human effort spent on the profile generation. In addition, this paper presents and discusses some experiments that are based on quantitative and qualitative evaluations.

Keywords: Systems Recommendation System, User Profile and Thematic Music

1 Introduction

Today, the songs can be electronically accessed as soon as they are published on the Web. The main advantage of open music is the minimization of the promotion time. In this context, Digital Libraries (DLs) have emerged as the main repositories of digital documents, links and associated metadata. The Recommender System involves information personalized. The personalization is related to the ways in which contents and services can be tailored to match the specific needs of a user or a community (Rarámuri people) [3]. The human-centered demand specification is not an easy task. One experiences this difficulty when trying to find a new song in a good indexing and retrieval system as a Web Radio.

The query formulation is complex and the fine tuning of the user requirements is a time-consuming task. Few users have enough time to spend some hours searching for, eventually new songs. This functionality, the query specification may be reached by the analysis of the user activities, history, information demands, in others. This paper presents a Musical recommendations system to a Web Radio; the songs recovered are

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 243-252

Received 17/10/09
Accepted 16/01/10
Final version 10/03/10

associated with Traditional Rarámuri Songs. The main contribution of this work is to provide a recommendation mechanism based on the user reducing the human effort spent on the profile generation. The paper is organized as follows. We start giving an overview of the background literature and concepts, then the recommender system and detail its architecture and techniques. Finally, we present some quantitative and qualitative experiments to evaluate and validate our system and discuss the results and conclusions of our work.

2 Bacground

The semantic Web technologies promote an efficient and intelligent access to the digital documents on the Web. The standards based on metadata to describe information objects have two main advantages: computational efficiency during the information harvesting process and interoperability among DLs. The first is a consequence of the increasing use of Dublin Core (DC) metadata standard [8]; the latter has been obtained as a result of the OAI initiative (Open Archive Initiative) [17]. DC metadata standard was conceived with the objective of defining a minimal metadata set that could be used to describe the available resources of a DL. This standard defines a set of 15 metadata (Dublin Core Metadata Element Set – DCMES) [8].

The main goal of OAI is to create a standard communication way, allowing DLs around the world to interoperate as a federation [21]. The DL metadata harvesting process is accomplished by the OAI-PMH protocol (Open Archives Initiative Protocol for Metadata Harvesting) [18], which define how the metadata transference between two entities, data and service providers, is performed. The data provider acts by searching the metadata in databases and making then available to a service provider, which uses the gathered data to provide specific services.

Considering that a Recommender System concerns with information personalization, it is essential that it copes with user profile. In our work, the user profile is obtained from a Web Radio similar at used in [13]. According to [11], there are three different methodologies used in Recommender Systems to perform recommendation: (i) content-based, which recommends items classified accordingly to the user profile and early choices; (ii) collaborative filtering, which deals with similarities among users' interests; and (iii) hybrid approach, which combines the two to take advantage of their benefits. In our work, the content-based approach is used, once the information about the user is taken from Web Radio users.

This recommendation process can be perceived as an information retrieval process, in which user's relevant songs should be retrieved and recommended. Thus, to perform recommendations, we can use the classical information retrieval models such as the Boolean Model, the Vector Space Model (VSM) or the Probabilistic Model [1, 9, and 20]. In this work, the VSM was selected since it provides satisfactory results with a convenient computational effort. In this model, songs and queries are represented by terms vectors. The terms are words or expressions extracted from the documents (lyrics) and from queries that can be used for content identification and representation. Each term has a weight associated to it to provide distinctions among

them according to their importance. According to [19] the weight can vary continuously between 0 and 1. Values near to 1 are more important while values near to 0 are irrelevant.

The VSM uses an n-dimensional space to represent the terms, where n corresponds to the number of distinct terms. For each document or query represented the weights represent the vector's coordinates in the corresponding dimension. The VSM principle is based on the inverse correlation between the distance (angle) among term vectors in the space and the similarity between the songs that they represented. To calculate the similarity score, the cosine (Equation 1) can be used. The resultant value indicates the relevance degree between a query (Q) and a document (song) (D), where w represents the weights of the terms contained in Q and D, and t represents the number of terms (size of the vector). This equation provides ranked retrieval output based on decreasing order of the ranked retrieval similarity values [19].

$$Similarity(Q,D) = \frac{\sum_{k=1}^{t} w_{qk} \cdot w_{dk}}{\sqrt{\sum_{k=1}^{t}(w_{qk})^2 \cdot \sum_{k=1}^{t}(w_{dk})^2}} \quad (1)$$

The same equation is widely used to compare the similarity among songs, and similarity, in our case, Q represents the user profile and D the documents descriptors (lyrics) that are harvested in the DL (see Section 3.2 for details). The term weighting scheme is very important to guarantee an effective retrieval process.

The results depend crucially of the term weighting system chosen, In addition, the query terms selection is fundamental to obtain a recommendation according to the user necessities. Our research is focused in the query terms selection and weighting. Any person with basic knowledge of Rarámuri Language which required a musical retrieval may evaluate the process complexity and the difficulty to find adequate songs. The central idea is to develop an automated retrieval and musical recommendation system where the price for the user is limited to the submission of an already existing preferences query.

3 The Recommender System

Our system focuses on the recommendation of Traditional Rarámuri songs. The information source to perform recommendations is the database associated with a Web Radio (Lyrics and Music of each song), while the user profile is obtained from Database Profile Register subset. However, any DL repository providing DC metadata and supporting the OAI-PMH protocol can be used as a source. An alternative to the user profile generation is under development. This alternative approach is composed by an information retrieval system to gather data from another Music sources. A DL repository stores digital songs or its localization (web or physical), and the respective metadata. A DL data provider allows an agent to harvest documents metadata through the OAI-PMH protocol. Our system handles the songs described with XML in DC standard [7, 15].

3.1 Recommendation System Architecture

In this section we present the architecture elements of our system and its functionalities (Fig. 1). To start the process, the users must supply their preferences in the XML version to the system. Whenever a user makes its registration in the system and sends his preferences list (1), the XML Songs Document module is activated and the information about the user's interests is stored in the local database named User Profile (2). Then the Metadata Harvesting module is activated to update the local database Songs Metadata. This module makes a request to a DL data provider to harvest specific documents metadata. It receives an XML document as response (3) and the XML DC to local DB module is activated (4). This module extracts the relevant metadata to perform the recommendation from the XML document and stores it in the local database named Songs Metadata (5). Once the user profile and the songs metadata are available in the local database, the Recommendation module can be activated (6). The focus is to retrieve lyrics and songs of a DL that the best matches the user profile described through the profile of each user in the Web Radio.

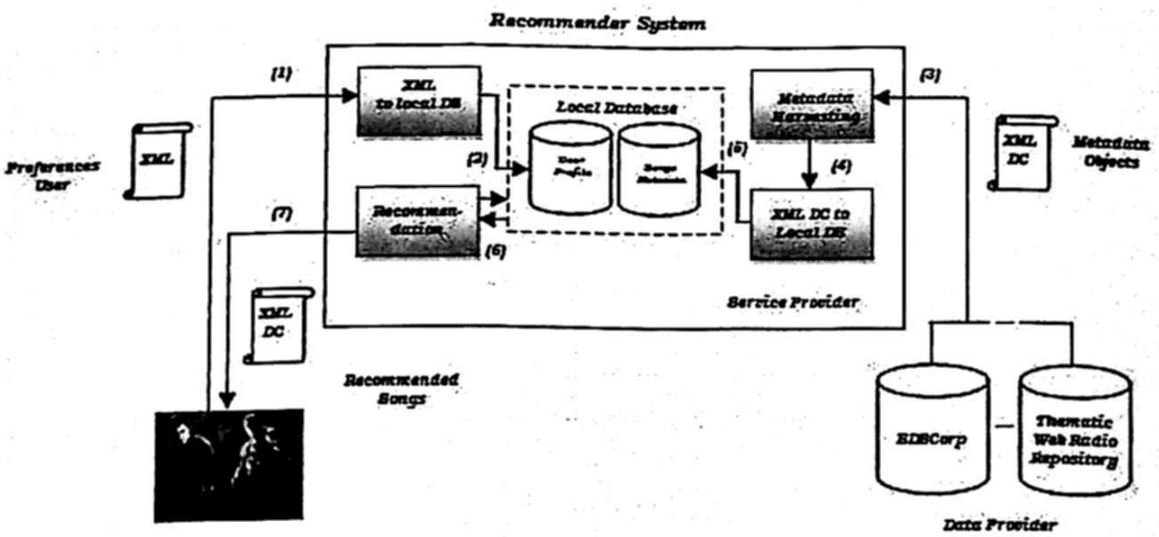

Fig. 1. The recommender system architecture.

3.2 The Recommendation Model

As stated before, the recommendation is based on the VSM model. The query vector is built with the term parsed from the title, keywords, singer or band and date. The parser ignores stop-words [5] (a list of common or general terms that are not used in the information retrieval process, e.g., prepositions, conjunctions and articles). The parser considers each term as a single word. On the other hand, the terms are taken integrally, as single expressions.

The query vector terms weights are build up according to the Equation 2. This equation considers the type of term (keyword or title), the language and the year of the first air data. Keyword terms are considered more important that the song's titles

and have more reading proficiency are more valorized (higher weight), and the terms obtained from the traditional Rarámuri songs are assigned with more important weight than translations to Rarámuri Language.

$$W_t = W_{KeywordOrTitle} * W_{Language} * W_{Year} \quad (2)$$

The weights $W_{KeywordOrTitle}$, $W_{Language}$, W_{Year} are calculated with Equation 3.

$$W_i = 1 - (i-1)\left[\frac{1 - w_{min}}{n-1}\right] \quad (3)$$

In this equation W_i varies according to the type of weight we want to compute. To illustrate this in the experimental evaluation (Section 4), for $W_{KeywordOrTitles}$ W_{min} was 0.95, and I is 1 if the language-skill.level is "good", 2 for "reasonable" and 3 for "few". For W_{Years} W_{min} was 0.55 and i vary from 1 to n, where n is the interval of years considered, begin 1 the highest and n the lowest. In the experimental evaluation it was considered a sample of songs between 2008 and 2002. However, if the interval is omitted, it will be considered as between the present year and the less recent year (the smallest between artist:first-song and artist:last-song).

If W_{min} is not informed, the default value will be used (presented in Equation 4). In the situation, Equation 3 is reduced to Equation 5.

$$W_{min\ default} = \frac{1}{N} \quad (4)$$

$$W_i = \frac{n - i + 1}{N} \quad (5)$$

Once the query vector is build, the songs vector terms and the respective weights must be defined. The adopted approach was (tf * idf), i.e., the product of the term frequency and the inverse document frequency [19]. This approach allows automatic term weights assignment for the songs retrieval. The term frequency (tf) corresponds to the number of occurrences of a term in a document. The inverse document frequency (idf) is a factor that varies inversely with the number of the songs n to which a term is assigned in a collection of N songs (typically computed as log (N/n)).

The best terms for content identification are those able to distinguish individuals ones from the remainder of the collection [19]. Thus, the best terms correspond to the ones with high term frequencies (tf) and low overall collection frequencies (high idf). To compute tf * idf, the system uses the DC metadata dc:title and dc:description to represent the songs content. Moreover, as your system deals with Ráramuri language and translation to Rarámuri Language, the total number of songs will vary accordingly. After building the query and songs vectors, the system is able to compute the similarities values among the songs and the query according to Equation 1.

4 Experimental Evaluation

In order to evaluate the musical recommender system, we have asked for preferences of listeners of a Web Radio. As response, a group of 57 people send us their list of preferences, whose information was loaded in the Songs Metadata related with the Web Radio local database. The songs Metadata local database was loaded in the User Profile local database related with it. This database stored up to August 2009, totalizing 87 songs from 11 singers or bands including in 7 albums.

After 20 recommendations were generated by the system for each hour in the Web Radio, considering individual's profile of the user and their preferences. This information was obtained using the user's data base related with the Web Radio.

Two evaluations were performed. The first was based on the hypothesis that the best songs to describe the profile of a user should be those produced by them. Since we had information about the songs by each user, we can match the items recommended to those. This evaluation was accomplished by the recall and precision metrics that is a standard evaluation strategy for information retrieval systems [1, 20]. The recall is used to measure the percentage of relevant songs retrieved in relation to the amount that should have been retrieved. In the case of songs categorization, the recall metric is used to measure the percentage of songs that are correct classified in relation to the number of songs that should be classified. Precision is used to measure the percentage of songs correctly recovered, i.e., the number of songs correctly retrieved divided by the number of songs retrieved.

As the profiles can be seen as classes and the songs as items to be classified in these profiles, we can verify the amount of items from the singers that are correctly identified (i.e. classified) by the user profile. As we have many users (i.e., many classes), it is necessary to combine the results. The macroaverage presented in Equation 6 was designed by D. Lewis [14] to perform this specific combination ("the unweighted mean of effectiveness across all categories"), and was applied by him in the evaluation of classification algorithms and techniques.

$$W_i \text{ macroaverage} = \frac{\sum_{i=1}^{n} X_i}{n} \tag{6}$$

In this formula, Xi is the recall or the precision, depending on the metric we want to evaluate, of each individual class (user in our case) and n is the number of classes (users). Thus, the macroaverage recall is the arithmetic average of the recalls obtained for each individual, and the macroaverage precision is the arithmetic average of the precisions obtained for each individual.

Given that the users are not interested in its own preferred songs as recommendations, we performed another evaluation that takes in to account only the items from other users. Then, 15 recommendations were presented to each individual ranked on the relative grade of relevance generated by the system. In this rank, the songs with the highest grade of similarity with the user profile were set as 100% relevant and the others were adjusted to a value relative to it. In this case, each author was requested to evaluate the recommendations generated to them assigning one of the following concepts (following the bipolar five-point Lickert scale); "Inadequate", "Bad", "Average", "Good", and "Excellent", and were also asked to comment the results. The following sections present the results obtained.

5 Analysis of Experiments

The first experiment was designed to evaluate the capability of the system to correctly identify the user profile (i.e., to represent its preferences), since we believe that the best songs to describe the user profile are those selected by themselves, as stated before. To perform such evaluation, we identified the songs of each user had at Web Radio. After that, we employed the recall metric to evaluate the number of songs recovered for each user and combined then with the microaverage equation explained before. We have found a macroaverage recall of 43.25%. It is important to state that each user received 20 recommendations. This is an acceptable value as the query construction was made automatically without human intervention. It happened to be lower than it should be if we have used more songs, maybe used translated songs, but the problem is the limited songs for singer or band. Other important consideration is that the recommendation ranking was generated with a depreciation degree that was dependent on year and language, as explained in the previous section. As the time-slice considered corresponds to period stored in the database related with the Web Radio, not all songs are good recommendations since the preferences changes along the time, similar at propose in [23].

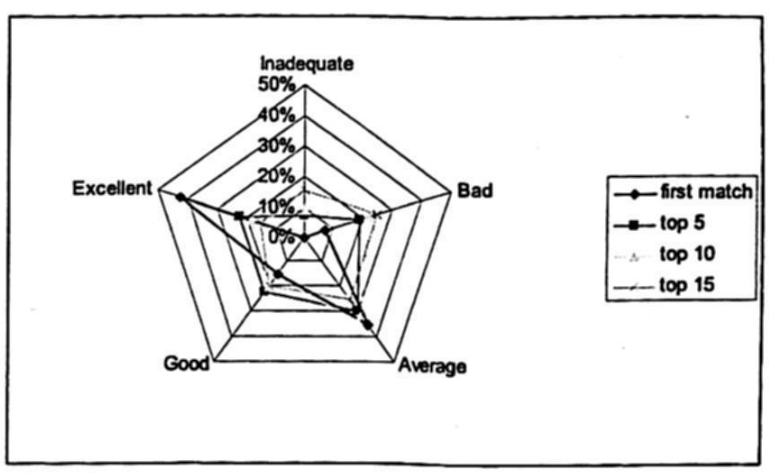

Fig. 2. Users' evaluations of the recommendations

Figure 2 presents the results of the second experiment, which was based on the users' qualitative evaluation of the recommended songs. On this experiment each user received 15 recommendations and evaluated them according to one of the following concepts: "inadequate", "bad", "average", "good", and "excellent". The results were grouped into the categories "first match", "top 5", "top 10", and "top 15", and are presented in Figure 2.

Analyzing three results, it is possible to observe that, if we only consider the first song recommended (the "first match"), the number of items qualified as "excellent" in greater than the others (i.e., 42.86%) and none of them were classified as "inadequate". This strengthens the capability of the system on performing recommendations adjusted to the present user's genre preferences interests. We have also grouped the concepts "good" and "excellent" into a category named "positive recommendation" and the concepts "bad" and "inadequate" into a "negative

recommendation" group, so we could obtain a better visualization and comprehension of the results (Fig. 3).

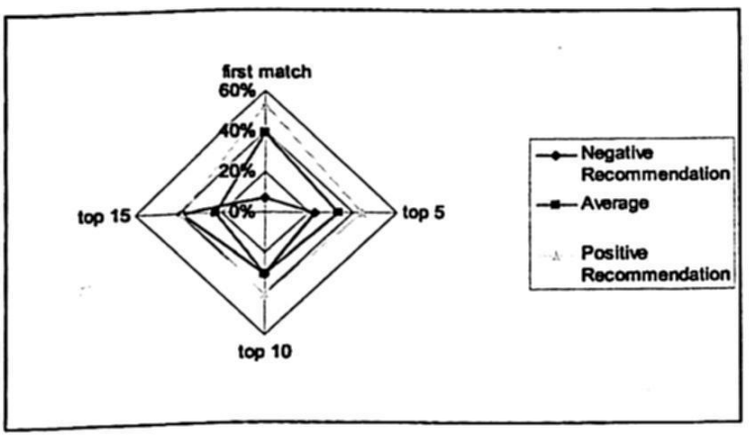

Fig. 3. Grouped users' evaluation

We could perceive that the positive recommendations, considering only the "first match", are superior (57.14%) in relation to the negative ones (7.14%). The same behavior can be perceived in the "top 5" and "top 10" categories, the recommendations had a negative evaluation only in the "top 15" category, and that probably happened because as the number of recommendations grows, the number of correct recommendations falls. It is clear that the automated procedure here adopted is adequate for an alert recommender system. Our proposal is to add to Web Radio an automated alert system that periodically sends to the user a list of the most relevant songs recently listen on it during seven or more weeks.

Further, in our tests the users that have changed their search in the last three months have negatively qualified the recommendations. In the next experiments a variable time threshold and different depreciation values will be employed and the temporal component will be exhaustively analyzed.

5 Conclusions

This paper presented a Recommender System to users of a Web Radio related with the lyrics of Rarámuri songs this dialect has 87500 speakers in Mexico (See Figure 4). In current days, in which the recovery of relevant digital information on the Web is a complex task, such systems are of great value to minimize the problems associated to the information overload phenomena, minimizing the time spent to access the right information.

The main contribution of this research consists on the heavy utilization of automated Musical Recommendation and in the use of a Digital Library (DL) metadata to create the recommendations. The system was evaluated with BDBComp, but it is designed to work with the open digital library protocol OAI-PMH, then it may be easily extended to work with any DL that supports this mechanism. The same occurs with the lyrics format related with the song, but it can be extended to support

other formats or to analyze information about the user. Alternatively operational prototype offers the possibility to the user to load the lyrics via an electronic form.

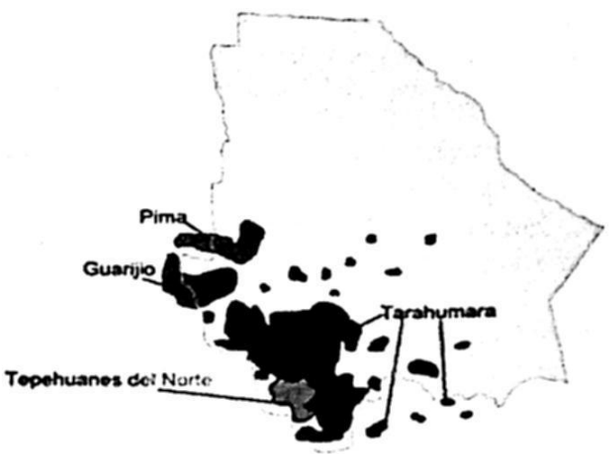

Fig. 4. Distribution of Rarámuri Language in Chihuahua State, México

The developed system will have many applications. One of them is the recommendation of songs to support a Web Radio related with Rarámuri music. Thus, the user could log into a specific thematic and receive recommendations of songs containing actualized relevant material to complement its current musical selection.

References

1. Baeza-Yates, R.; Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley, Workingham, UK (1999)
2. BDBComp Biblioteca Digital Brasileira de Computação. Online at: http://www.lbd.dcc.ufmg.br/bdbcomp/ (last access: 2006-11)
3. Callahan, J. et al.: Personalization and Recommender Systems in Digital Libraries. Joint NSF-EU DELOS Working Group Report, May (2003)
4. CITIDEL: Computing and Information Technology Interactive Digital Educational Library. Institut interfacultaire d'informatique, University of Neuchatel. Online at: http://www.unine.ch/info/clef/ (last access: 2005)
5. CLEF and Multilingual information retrieval. Institut interfacultaire d'informatique, University of Neuchatel. http://www.unine.ch/info/clef/ (last access: 2005)
6. Contessa, D., Fraga F. and Palazzo A.: An OAI Data Preovider for JEMS. Proceedings of the ACM DocEng 2006 Conference, Amsterdam. Oct (2006) 218-220
7. DC-OAI: A XML schema for validating Unqualified Dublin Core metadata associated with the reserved oai_dc metadataPrefix. Online at: http://www.openarchives.org/OAI/2.0/oai_dc.xsd (last access: 2005)
8. Dublin Core Metadata Initiative. http://dublincore.org (last access: 2005)
9. Grossman, David A. Information retrieval: algorithms and heuristics. 2nd ed. Dordrecht: Springer, (2004) 332
10. Gutteridge, C. GNU EPrints 2 overview, Jan. 01 (2002).
11. Huang, Z. et al. A Graph-based Recommender System for Digital Library. In JCDL'02 Portland, Oregon (2002)

12. Laender, A., Gonçalves, M. and Roberto, P.: BDBComp: Building a Digital Library for the BrazilianComputer Science Community. In Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries, Tucson, AZ; USA (2004) 23-24
13. Laustanou, K.: MySpace Music (2007)
14. Lewis, D.D.; Evaluating text categorization. In Proceedings of Speech and Natural Language Workshop. Defense Advanced Research Projects Agency, Morgan Kaufmann. (1991) 312-318
15. LPMP-CNPq. Padronização XML: Curriculum Vitae. Online at: http://lmpl.cnpq.br. (last access: 2005-03)
16. Maly, K., Nelson, M., Zubaír, M., Amrou, A, Kothamasa, S., Wang, L. and Luce, R.: Lightweight communal digital libraries. In Proceedings of JCDL'04, Tucson; AZ (2004) 237-238
17. OAI: Open Archives Initiative. Online at: http://openarchive.org (last access: 2005-10)
18. OAI-PMH. The Open Archives Initiative Protocol for Metadata Harvesting. Online at: http://www.openarchives.org/OAI/2.0/openarchivesprotocol.htm (last access: 2005-11)
19. Salton, G. and Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval, Information Processing and Management an International Journal, v.24, Issue 5, (1988) 513-523
20. Salton, G. and Macgill, M.: Introduction to Modern Information Retrieval. New York. McGraw-Hill. (1983) 448
21. Sompel, H. and de Lagoze, C.: The Santa Fe Convention of the Open Archives Initiative D-Lib Magazine, [S.1.], v.6, n.2, Feb (2000)
22. Tansley, R., Bass, M., Stuve, D., Branschofsky, M., Chudnov, D., McClellan, G. and Smith, M.: Dspace: An institutional digital repository system. In Proceedings of JCDL'03, Houston, TX (2003) 87-97
23. Ochoa, A et al. Musical Recommendation on Thematic Web Radio. In Journal of Computers, Oulu, Finland (2009) By publishing

Improving Clustering of Noisy Documents through Automatic Summarisation

Seemab Latif[1], Mary McGee Wood[2], and Goran Nenadic[1]

[1] School of Computer Science
University of Manchester, UK
latifs@cs.man.ac.uk,G.Nenadic@cs.man.ac.uk
[2] Assessment21, Cooper Buildings
Sheffield, UK
mary@cs.man.ac.uk,mmw@assessment21.com

Abstract. In this paper we discuss clustering of students' textual answers in examinations to provide grouping that will help with their marking. Since such answers may contain noisy sentences, automatic summarisation has been applied as a pre-processing technique. These summarised answers are then clustered based on their similarity, using k-means and agglomerative clustering algorithms. We have evaluated the quality of document clustering results when applied to full-texts and summarized texts. The analyses show that the automatic summarization has filtered out noisy sentences from the documents, which has made the resulting clusters more homogeneous, complete and coherent.

1 Introduction

Document clustering is a generic problem with widespread applications. The motivation behind clustering a set of documents is to find inherent structure and relationship between the documents. Document clustering has been applied successfully in the field of Information Retrieval, web applications to assist in the organization of the information on the web (Maarek et al., 2000), to cluster biomedical literature (Illhoi et al., 2006) and to improve document understanding by using document clustering and multiple document summarisation where summarisation is used to summarise the documents in resultant clusters (Wang et al., 2008).

The quality of document clustering is dependent on the length of the documents and the "noise" present within documents. The main features of a document are the words that it contains. Unimportant words contribute to noise and thus may lead to faulty results from automatic processing. Assess By Computer (ABC) (Sargeant et al., 2004) is a suit of software tools for assessment of students' exams. It uses document clustering to group students' textual answers to support semi-automated marking and assessment in examinations (Wood et al., 2006). One important feature of ABC is abstraction: instead of referring to a large number of students' answers one at a time, ABC refers to them as groups of similar answers at one time. Students' answers are written under time pressure in an examination environment. Therefore, there is always a high chance of students making spelling mistakes and writing sentences or words that are not relevant

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 253-264

Received 02/12/09
Accepted 16/01/10
Final version 12/03/10

to the question being asked and thus could be considered as noise, in particular, if we want to assess students' understanding of the concepts in questions. These sentences or words may affect the performance of the clustering process. To avoid the deterioration of the clustering results, we hypothesized that Automatic Text Summarisation could be one of the solutions to remove noisy data from the answers efficiently and effectively.

Previous extensive work on document clustering has focused on issues such as initial number of clusters, similarity measures and document representation, and has to some extent ignored the issue of document pre-processing. The principal pre-processing techniques applied to documents have been stopword removal, stemming and case folding. However, document clustering algorithms are highly dependent on the length of the document (Hsi-Cheng and Chiun-Chieh, 2005). Therefore, we hypothesize that pre-processing texts using automatic summarization before document clustering may improve the performance of clustering algorithms in terms of both efficiency and quality of the clustering, and summary based clusters will be more homogeneous, complete and coherent than fulltext clusters.

2 Integrating Text Summarisation into Document Clustering

In this paper, we integrate text summarisation with document clustering to utilize the mutual influence of both techniques to help in grouping of students' free text answers. We apply automatic text summarisation as a pre-processing method for students' answers before clustering. The main aim behind using automatic text summarisation is to extract the information content of the answers, and to improve document clustering by reducing the noise and the length of the answers.

2.1 Automatic Summarisation

In this paper, we have used Keyword-dependent Summarizer (KdS) that summarises a document using keywords given by the user. In case of summarising students' answers, these keywords are given by the marker. This summarisation method follows the shallow approach for summarization, i.e. it employs text extraction techniques to generate the summary (Neto et al., 2002). Abstractive or language generation approach for summarisation was not used, as there is no guarantee that generated the text will be useful (Neto et al., 2000). In order to perform extractive summarisation, the first question arises is that on what granularity segments will be extracted i.e. the "size" of the textual units that are copied from the original text and pasted into the summary. This could be a paragraph, a sentence, a phrase or a clause. We have performed summarisation on sentence level using a structural analysis technique. This technique uses the semantic structure of the text (such as keyword frequency, position of the sentence in the document and paragraph, cue phrases and sentence length) to rank sentences for inclusion in a final summary.

This summarization method has been evaluated qualitatively and quantitatively by both human and automatic evaluations. Both human and ROUGE ((Lin and Hovy, 2003)) evaluations have achieved high positive correlation with the scores assigned to the essays by the human marker.

2.2 Clustering process with Summarisation: Framework

Clustering students' free text answers is a difficult task as there are a number of problems posed by natural language for automated processing of these answers. In this paper, each student answer is referred to as a document. Documents are represented as a vector of words in the Vector Space Model (VSM). The performance of the VSM is highly dependent on the number of dimensions in the model and it can be improved by removing terms that carry less semantic meaning and are not helpful while calculating the similarity between documents (stopword removal), converting terms to their stems (stemming), spelling corrections and term weighting (the process of modifying the values associated with each term to reflect the information content of that term). The similarity between document vectors is calculated using the cosine of the angle between them.

The overall framework of the integration of text summarisation with document clustering for clustering students' answers is given in Figure 1 and it follows the following steps:

- **Step 1: Extracting Answers**
  In this step, students' answer strings for a question, which are to be clustered, are extracted from the XML file provided by the ABC marking tool.

- **Step 2: Summarising Answers**
  Each answer string is then summarised using KdS. Summarisation extracts the information content of the answers and filters out the noise. This reduces the number of terms in the answer string.

- **Step 3: Document Pre-processing**
  The summarised answers are then pre-processed using spelling correction, stopword removal, stemming and term weighting. First three pre-processing steps and summarisation aim to reduce the total number of terms used in clustering while term weighting provides a way to apply different weights to the terms to reflect their importance.

- **Step 4: Creating the Term-By-Document Matrix**
  The term-by-document matrix on full-text answer strings could potentially be a sparse matrix containing over 500 terms (dimensions). Many terms in the matrix are noisy and useless, giving no semantic information about the answer. To improve the performance of the term-by-document matrix, we have created vectors on the summarised answer strings instead of full-text answer strings. The matrix created on summarised answer strings has fewer dimensions as compared to the

matrix crated on full-text answer strings.

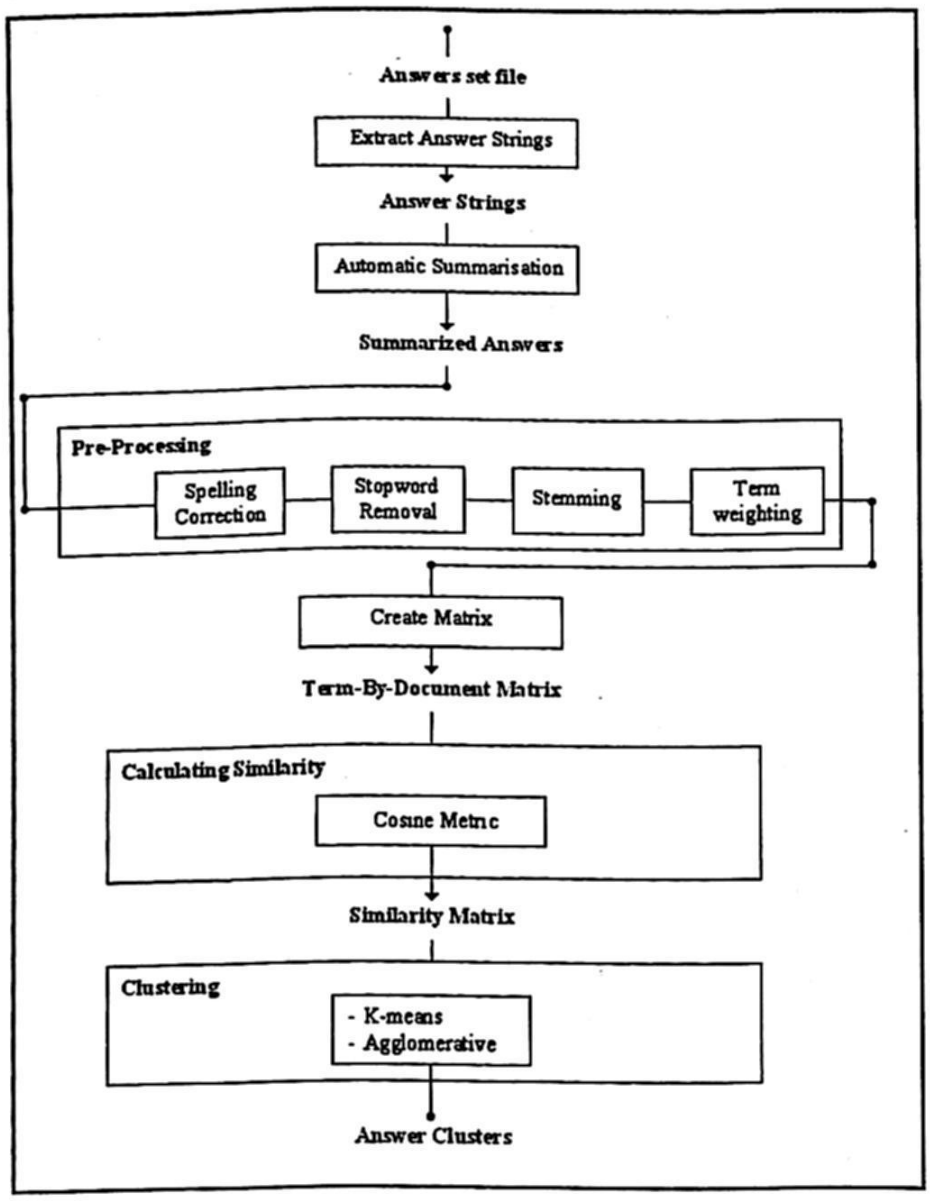

Fig. 1. Framework for Clustering Students' Answers

– **Step 5: Creating Clusters**
The last step in this procedure is to create clusters. Two clustering algorithms are used to generate clusters. Clustering algorithms cluster answers based on the terms present within the answer and the similarity between answers.

3 Experimental Setup

In this section, the design and results of an experiment are discussed. Here, automatic summarisation has been applied as a pre-processing step for document clustering to reduce the noise. The full documents and summarised documents are then clustered and clustering results are evaluated using the precision and recall measures (defined below).

The aim of this experiment is to evaluate the clustering results, when carried on full-text and summarised documents, for their quality and accuracy.

Experimental Design: The automatic summarisation pre-processing method was applied for partitional (k-means) and hierarchical (agglomerative) clustering algorithms. We have used KdS and the keywords were taken from the "Model Answer" to the examination questions. The clustering algorithms were run on the full-text documents and on five different levels of summarisation compression. Each document in the dataset was compressed to 50%, 40%, 30%, 20% and 10% of the size of its original length, and both algorithms were run on the documents in each dataset having five different levels of compressed documents and the whole full-text document.

Datasets: The datasets used for this experiment were taken from the ABC marking tool, and include marked exams from Pharmacy, Biology and Computer Science undergraduate courses conducted at the University over the past five years. The reason for taking these datasets is that human assigned marks for these datasets are available and these marks will be used to evaluate the clustering results. These exams were marked by an instructor who was teaching the course.

Table 1 gives the number of answers in each dataset, average number of terms in the answers and the number of clusters generated for each dataset. The number of clusters for each dataset was the total number of distinct marks for that question plus one (for mark 0). For example, if we want to cluster answers to a question worth of 4 marks then the number of clusters will be 5 (one for mark 0). As a sample, one question along with its answer from the Biology and Pharmacy datasets is given in appendix A.

Table 1. Datasets statistics for evaluating summary-based clusters

Dataset	Question ID	Number of Documents	Number of Features	Number of Clusters
CS	CS1	108	880	5
Biology	Biology1	279	960	6
	Biology2	280	708	6
	Biology3	276	909	7
	Biology4	276	996	6
Pharmacy	Phar1	139	1103	5
	Phar2	165	1278	7

3.1 Evaluation Metrics

In general, the categories are not known before hand, but in our case, we have used human-marked data from the ABC marking tool to evaluate the quality of clustering.

The answers in each dataset were grouped according to their marks awarded by the human marker. These marked categories have served the purpose of the "gold standard" categories for the evaluation. For the evaluation of document clustering results, Precision and Recall were defined as follows.

Precision associated with one document represents what fraction of documents in its cluster belongs to its marked category. Precision of a clustering result on a dataset can be either calculated at the document level (micro-precision) or at the cluster level (macro-precision). If the micro-precision is high, then it means that the number of noisy documents or misclassified documents in the clusters is low. If macro-precision is high then it means that the most documents from the same category are grouped together.
Recall associated with one document represents what fraction of documents from its marked category appears in its cluster. Similarly to precision, recall of a clustering result on a dataset can be either calculated on the document level (micro-recall) or on the cluster level (macro-recall). If the micro-recall is high then it means that most of the documents from its model category lie in its cluster. If macro-recall is high then it means clusters are similar to the model categories.

We have used only micro-precision and micro-recall measures for clustering result evaluations, which are combined using the standard combining metric, Van Rijsbergen's F-Measure (Rijsbergen, 1974). Both precision and recall of a cluster will be high when the computed cluster is similar to the model cluster. The precision will be 1 and the recall of each document will decrease by $\frac{1}{|ModelCategory|}$ when each document is placed in independent cluster i.e. each cluster has only one document. The recall will be 1 and the precision of each document in the cluster will decrease by $\frac{n}{n+1}$ when all the documents will be clustered into one cluster. Note that it is not possible to have precision or recall of 0 because at least a cluster and a category share one common document.

There are four mathematical constraints proposed by (Amigo et al., 2009) that should be satisfied by the clustering evaluation metrics. These constraints are Cluster Homogeneity, Cluster Completeness, Rag Bag and Cluster Size versus Quantity. The metrics that we have introduced here satisfy these constraints. Micro-precision satisfies cluster homogeneity and ragbag constraints while micro-recall satisfies cluster completeness and size versus quantity constraints.

4 Results and Analysis

Because of random initialization of the k-means, the algorithm was run 10 times on each dataset and then mean values were calculated for the evaluation. All summarisation-based clustering results have performed better then the full-text clustering results. In tables 2 and 3, the micro-precision values are higher for the summarisation-based clusters than the micro-precision values for the full-text clusters.

Table 2. Micro-precision for k-means clustering

Dataset	Question ID	Fulltext	Compression Level				
			50%	40%	30%	20%	10%
CS	CS1	0.513	0.831	0.837	0.793	0.753	0.728
Biology	Biology1	0.519	0.843	0.880	0.838	0.776	0.711
	Biology2	0.495	0.904	0.899	0.852	0.832	0.780
	Biology3	0.617	0.814	0.823	0.759	0.711	0.677
	Biology4	0.672	0.777	0.786	0.764	0.719	0.679
Pharmacy	Phar1	0.569	0.774	0.800	0.737	0.691	0.642
	Phar2	0.586	0.798	0.811	0.770	0.720	0.663

Table 3. Micro-precision for agglomerative clustering

Dataset	Question ID	Fulltext	Compression Level				
			50%	40%	30%	20%	10%
CS	CS1	0.466	0.840	0.840	0.783	0.750	0.694
Biology	Biology1	0.715	0.855	0.862	0.780	0.805	0.727
	Biology2	0.738	0.900	0.895	0.865	0.849	0.736
	Biology3	0.673	0.809	0.843	0.764	0.718	0.675
	Biology4	0.695	0.786	0.800	0.753	0.707	0.705
Pharmacy	Phar1	0.599	0.775	0.779	0.762	0.718	0.602
	Phar2	0.655	0.772	0.812	0.761	0.721	0.678

According to the definition of micro-precision, it is high when most of the items from a single model category are clustered together in one cluster or when majority of the items have their individual clusters. In our case, the numbers of clusters are fixed for each dataset, which omits the possibility of each document having its own cluster. This means that most of the documents belonging to a single model category are clustered together and that summarisation has filtered out the feature terms that are not useful in distinguishing the documents. Therefore, summary-based clusters are more homogeneous and noise free than full-text clusters.

Micro-recall values for each of the algorithms are given in tables 4 and 5. These values are high for full-text clusters. According to the definition of micro-recall, it is high when the resultant clusters are similar to the model categories or when majority of the items are clustered in one cluster. We analysed the full-text document clustering results manually, which showed that the high micro-recall for full-text clusters is due to the clustering of most of the documents in one cluster.

However, many of the feature terms in full-text documents have low distinctive power and are not useful in clustering. This suggests that if initial documents are misclassified then these documents "attract" other documents with high similarity and a large number of common feature terms. This will make the cluster noisy with the doc-

Table 4. Micro-recall for k-means clustering

Dataset	Question ID	Fulltext	Compression Level				
			50%	40%	30%	20%	10%
CS	CS 1	0.795	0.715	0.769	0.678	0.592	0.549
Biology	Biology1	0.688	0.724	0.730	0.838	0.616	0.545
	Biology2	0.769	0.668	0.715	0.643	0.588	0.518
	Biology3	0.674	0.760	0.769	0.706	0.655	0.618
	Biology4	0.697	0.772	0.772	0.750	0.702	0.650
Pharmacy	Phar1	0.628	0.756	0.764	0.695	0.653	0.590
	Phar2	0.638	0.754	0.771	0.725	0.667	0.606

Table 5. Micro-recall for agglomerative clustering

Dataset	Question ID	Fulltext	Compression Level				
			50%	40%	30%	20%	10%
CS	CS1	0.863	0.727	0.768	0.712	0.556	0.564
Biology	Biology1	0.574	0.733	0.733	0.722	0.620	0.599
	Biology2	0.496	0.645	0.680	0.622	0.586	0.530
	Biology3	0.613	0.74	0.777	0.727	0.675	0.636
	Biolog 4	0.667	0.781	0.779	0.739	0.684	0.664
Pharmacy	Phar1	0.557	0.713	0.728	0.683	0.690	0.558
	Phar2	0.598	0.764	0.761	0.713	0.676	0.617

uments that are not part of it.

Tables 6 and 7 and figures 2 and 3 give the micro-F-measure values for the k-means and agglomerative clusterings on the three datasets.

Analyses of the results show that both clustering algorithms results have achieved optimal clustering at the compression level of 40% summarised documents. At this level, micro-F-measure is $\simeq$ to 80% for both algorithms, averaged over all the datasets. The datasets with large number of documents have a smooth curve with peak at 40% summary based clustering. For most of the datasets even 10% summarised text clustering has performed better than the full-text clustering.

5 Conclusion

In this paper, the experiments have been discussed which evaluate the method of improving clustering results by performing automatic summarisation of students' textual answers. Automatic summarisation has reduced the length of the documents and hence the number of feature terms that were potentially noisy for clustering. The results suggest that automatic summarisation has filtered out the noise from the documents and

Table 6. Micro-F-measure for k-means clustering

Dataset	Question ID	Fulltext	Compression Level				
			50%	40%	30%	20%	10%
CS	CS1	0.624	0.768	0.802	0.731	0.662	0.626
Biology	Biology1	0.591	0.778	0.797	0.741	0.685	0.616
	Biology2	0.602	0.767	0.796	0.733	0.689	0.622
	Biology3	0.644	0.786	0.795	0.731	0.682	0.646
	Biology4	0.684	0.774	0.779	0.757	0.690	0.664
Pharmacy	Phar1	0.597	0.764	0.782	0.715	0.672	0.614
	Phar2	0.611	0.775	0.790	0.746	0.693	0.633
			Average	0.791			

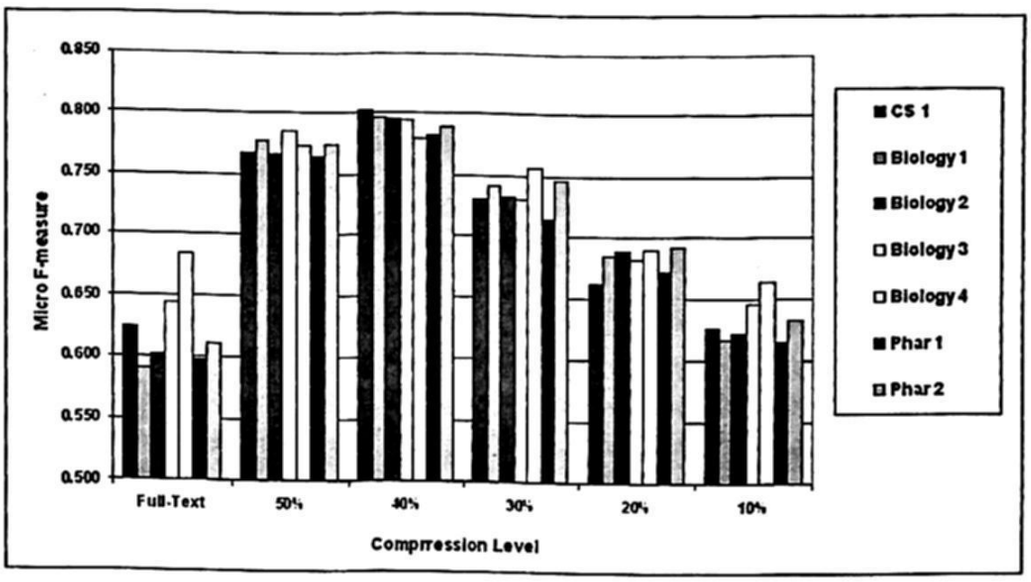

Fig. 2. K-means clustering micro-F-measure

Table 7. Micro-F-measure for agglomerative clustering

Dataset	Question ID	Fulltext	Compression Level				
			50%	40%	30%	20%	10%
CS	CS1	0.605	0.780	0.802	0.746	0.638	0.623
Biology	Biology1	0.637	0.789	0.792	0.750	0.701	0.657
	Biology2	0.593	0.751	0.773	0.723	0.694	0.616
	Biology3	0.641	0.773	0.808	0.745	0.696	0.655
	Biology4	0.681	0.784	0.789	0.746	0.695	0.684
Pharmacy	Phar1	0.577	0.743	0.753	0.720	0.703	0.579
	Phar2	0.625	0.768	0.786	0.736	0.698	0.646
			Average	0.790			

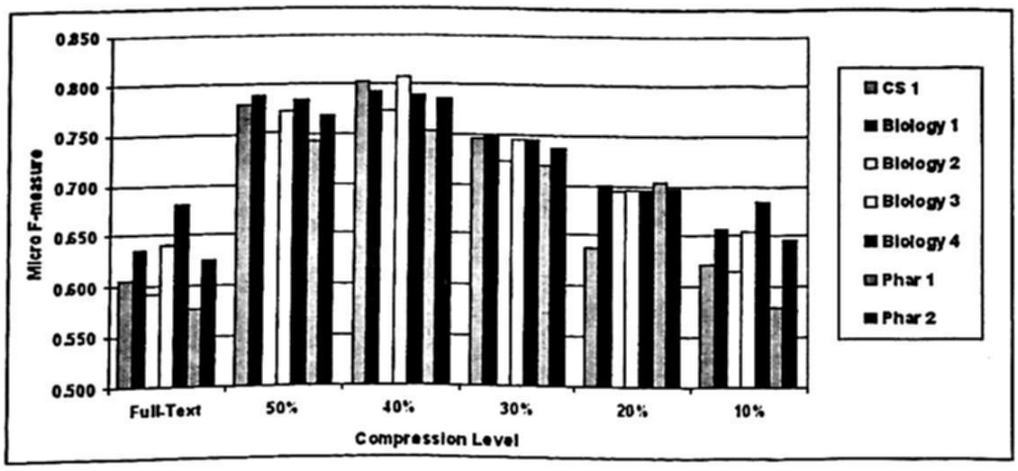

Fig. 3. Agglomerative clustering micro-F-measure

has extracted the relevant information content of the documents.

Due to the noise removal and reduction in document length, the performance of the two document clustering algorithms has improved in terms of quality of the clustering results. While evaluating the clustering results, we came to conclusion that summarisation-based clusters are more homogeneous (as micro-precision of summary-based clusters is higher than the micro-precision of full-text clusters for both the algorithms, see tables 2 and 3) and complete (as micro-recall value of summary-based clusters is higher than the micro-recall for full-text clusters except for 10% summarised-text clusters using k-means, see tables 4 and 5) then full-text clusters. Optimal clustering results were achieved when the documents were summarised to 40% of their original length.

References

1. Amigo, E., Gonzalo, J., Artiles, J. and Verdejo, F. (2009). A Comparison of Extrinsic Clustering Evaluation Metrics based on Formal Constraints, *Information Retrieval Journal* 12(4): 461– 486.
2. Hsi-Cheng, C. and Chiun-Chieh, H. (2005). Using Topic Keyword Clusters for Automatic Doc- ument Clustering, *Proceedings of the 3rd International Conference on Information Technol- ogy and Applications*, pp. 419–424.
3. Illhoi, Y., Hu, X. and Il-Yeol, S. (2006). A Coherent Biomedical Literature Clustering and Sum- marization Approachthrough Ontology-Enriched Graphical Representations, *Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery*, pp. 374– 383.
4. Lin, C. and Hovy, E. (2003). Automatic Evaluation of Summaries using N-gram Co-occurrence Statistics, *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pp. 71–78.
5. Maarek, Y., Fagin, R., Ben-Shaul, I. and Pelleg, D. (2000). Ephemeral Document Clustering for Web Applications, *Technical Report RJ 10186*, IBM Research Report.
6. Neto, J., Freitas, A. and Kaestner, C. (2002). Automatic Text Summarization using a Machine Learning Approach, *Proceedings of the 16th Brazilian Symposium on Artificial Intelligence, Advances in Artificial Intelligence*, pp. 205–215.

7. Neto, L., Santos, A., Kaestner, C. A. and Freitas, A. (2000). Document Clustering and Text Summarization, *Proceedings of the 4th International Conference on Practical Applications of Knowledge Discovery and Data Mining*, pp. 41–55.
8. Rijsbergen, C. V. (1974). Foundation of Evaluation, *Journal of Documentation* **30**(4): 365–373.
9. Sargeant, J., Wood, M. and Anderson, S. (2004). A Human-Computer Collaborative Approach to the Marking of Free Text Answers, *Proceeding of the 8th International Conference on Computer Assissted Assessment*, pp. 361–370.
10. Wang, D., Zhu, S., Li, T., Chi, Y. and Gong, Y. (2008). Integrating Clustering and Multi-Document Summarization to Improve Document Understanding, *Proceeding of the 17th Conference on Information and Knowledge Management*, pp. 1435–1436.
11. Wood, M., Jones, C., Sargeant, J. and Reed, P. (2006). Light-Weight Clustering Techniques for Short Text Answers in HCC CAA, *Proceedings of the 10th International Conference on Computer Assisted Assessment*, pp. 291–305.

Appendix A

One question along with its answer from Biology and Pharmacy dataset is given in this appendix.

Biology II

Question: What do you understand by the term Haematocrit? Could a person have a normal RBC count but a low Haematocrit? What could be the cause of this?

Answer: The Haematocrit shows the relative proportion of red blood cells to plasma and (white blood cells and other proteins), the figure given being the proportion of 'packed red blood cells'. The ideal proportion of packed cell volume (haematocrit) being, 37-47% in females, and 40-54% in males. If a person was to have a normal red blood cell count (normal amount of cells per micro litre), but smaller cells (microcytic cells) due to a disorder with the peptide chains, for example a missing peptide chain, then the cells would be able to pack closer together, and hence have a low haematocrit. This is classed as microcytic anemia, and can be due to problems in transcribing the two alpha or two beta chains, or perhaps the inability to form the globular protein. Eitherway, the size of the haemoglobin is reduced, hence the volume is reduced, but there is still likely to be the same number of cells. Other reasons for a normal red blood count and a low Haematocrit could be an increase in bodily fluids, i.e. plasma. The red blood cell count would be considered within 'normal' range, however, due to the increase in plasma, the ratio of blood to plasma would be altered in that the Haematocrit value would be lower.

Pharmacy

Question: State the factors to be considered when selecting an antiepileptic regimen for an 18 year old female patient who has been newly diagnosed as suffering from epilepsy by the local Neurologist.

Answer: Firstly, before selecting a treatment regimen, it is important to establish the number of seizures the patient has suffered. This is because treatment is rarely initiated after a single seizure. Usually, a person must suffer from two seizures in twelve months before treatment is given. The type of seizure is also important in selecting a treatment regimen i.e partial seizures are generally treated differently to general seizures. These classes of seizure can be further subdivided and will have specific treatment protocols. Thirdly, it is important to establish whether the patient is taking any other medication or has any other medical conditions which could affect the treatment options. For example, many antiepileptic drugs can interact with and reduce the efficacy of the combined oral contraceptive. It is important to try and give the patient a single agent wherever possible as many antiepileptic patients are successfully controlled with monotherapy. This would avoid the problems asociated with polypharmacy such as drug interactions, increased drug tox-

icity and reduced compliance. The age of the patient also needs to be considered especially if they were very young or elderly as this may limit the treatment options available or the doses may need adjusting. This does not seem to be a problem in this patient as she is 18 years old. However, the fact that she is a female needs to be considered. This is because many of the drugs can cause unacceptable side effects in women i.e. sodium valproate can cause hair loss, phenytoin can cause acne and gingival hyperplasia. This woman is also of child bearing age and it would need to be established if the patient was pregnant or breastfeeding as many of the antiepileptic drugs are teratogenic. It would also need to be established whether this patient was taking the combined oral contraceptive pill as antieplieptic medication can reduce the efficacy of this. This would mean that other precautionary advise on alternative methods of contraception would need to be given.

Educational Applications

User Profile Modeling in eLearning using Sentiment Extraction from Text

Adrian Iftene and Ancuţa Rotaru

Faculty of Computer Science, "Al. I. Cuza" University,
General Berthelot Street, No. 13, 700483 Iasi, Romania
{adiftene, ancuta.rotaru}@infoiasi.ro

Abstract. This paper addresses an important issue in the context of current Web applications because there are new ideas of providing personalized services to users. This part of web applications is a very controversial one because it is very hard to identify the main component which should be emphasized, namely, the development of a user model. Customizing an application has many advantages (the elimination of repeated tasks, behavior recognition, indicating a shorter way to achieve a particular purpose, filtering out irrelevant information for an user, flexibility), but also disadvantages (there are users who wish to maintain anonymity, users who refuse the offered customization or users who do not trust the effectiveness of personalization systems). This allows us to say that in this field there are many things to be done: the personalization systems can be improved; the user models which are created can be adapted to a larger area of applications, etc. The eLearning system created by us has as an aim to reduce the distance between the involved actors (the student and the teacher), providing easy communication using the Internet. Thus, based on this application students can ask questions and teachers can provide answers to them. Then, on the basis of this dialogue using the sentiment extraction from text, we built a user model in order to improve the communication between the student and the teacher. Therefore we built a user's model, but we helped the teacher to understand better the problems faced by the students.

Keywords: eLearning, Sentiment Extraction, User Profile Modeling

1 Introduction

Web pages are customized for specific users based on certain features such as: interests, the social class they belong or the context in which they access the pages. Customizing itself starts with creating a user model that includes modeling skills and the user's knowledge. This model can predict (as appropriate) normally carried out mistakes during the learning process because it is basically a collection of user's personal data.

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 267-278

Received 23/11/09
Accepted 16/01/10
Final version 09/03/10

The models of users are found in social Web and are used to describe how people socialize and how they interact via the World Wide Web. For example, people can explicitly define their identity by creating a profile in social networking services like Facebook, LinkedIn and MySpace or tacitly by creating blogs to express their own opinions.

GNU Mailman[1] is free software for managing electronic mail discussion and e-newsletter lists. Mailman is integrated with the web, making it easy for users to manage their accounts and for list owners to administer their lists. Mailman supports built-in archiving, automatic bounce processing, content filtering, digest delivery, spam filters, digests (RFC 934 and RFC 1153), Usenet gateways and more. It provides a web front-end for easy administration, both for list owners and list members.

Our system is similar to the one presented before, but in addition we have approached the subject to achieve a user's model in the eLearning system and have exhibited different traits that underlie it, such as interests, knowledge, experience, goals, personal traits and work context. In addition, we describe the created patterns, the forms of representation (e.g., vectors or arrays of concepts), but we also discuss about the recommendation systems (systems that give users some recommendations depending on the created model or depending on their interests).

For creating the user model we used a module for extracting the sentiment from Romanian texts similar to [1], and so we emphasized the positive (*frumos* – En: *nice*, *excelent* – En: *excellent*), negative (*groaznic* – En: *awful*, *teamă* – En: *fear*) or neutral (*a merge* – En: *to go*, *masă* – En: *weight*) significance of terms from a text. Also, we identify the role of negation (*nu* – En: *not*, *niciodată* – En: *never*), of diminutive words (*mai puțin* – En: *less*, *probabil* – En: *probable*) and of intensification words (*desigur* – En: *sure*, *cert* – En: *certain*). To investigate the significance of each term we used Romanian WordNet[2] and after that we created our own resource with specific terms. Thus, we first add general terms about people, taken from Maslow's pyramid of human motivation[3] [6], then we add for each of these terms its synonym, hyponym, hypernym and after that we add to our custom resource the field's terms for which we built this system.

The application is a site created for second year students from our faculty (Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iasi), where they could ask the teachers questions on certain subjects. The purpose of this application is to contribute to the strengthening of teacher-student relationship through ease of communication between them in moments when they cannot meet for various reasons. Of course, for each of the two types of users who interact with the application we provide specific functionality. Thus, a student may ask questions with a specific purpose stated explicitly or with a certain priority and can view the answers provided by the teacher, while a teacher can answer questions and can view a full report of each student's work on the site.

Using this theme (the teacher-student communication) we built a database of features which was modeled by our system for each student. To assess the quality of

---

[1] GNU Mailman: http://www.gnu.org/software/mailman/index.html

[2] WordNet: http://wordnet.princeton.edu/

[3] Maslow hierarchy: http://en.wikipedia.org/wiki/Maslow%27s_hierarchy_of_needs

these results on the one hand and to improve the quality of the recommendation system on the other hand, we created two psychological tests which we have offered for students to complete. Finally, based on this information we were able to formulate a recommendation for teachers in order to improve the communication with students.

2 eLearning Elements

Both teachers and psychologists have concluded that the activity of learning is a process which involves all the components of human personality, such as from the simplest to the most complex targets the entire human system is put in motion in order to achieve the reception, the processing, the interpretation and the exploitation of the information.

The eLearning concept involves combining art and psycho-pedagogy because the starting point in building such a system is the knowledge of learning mechanisms (in order to work more efficiently, so as to assimilate through these technical systems).

eLearning[4] (Electronic Learning) is a type of technology supporting education in which training takes place via computer technology. This form of education is used in many areas: in companies to provide training courses for employees, in universities to define a specific mode to attend a course (especially when students do not take any courses to date).

Although it is said it does not favors face-to-face interaction (the user uses only the computer to find the necessary information) as opposed to the standard education where there is always one teacher who teaches the student, this new educational technology can be used in combination with already established techniques: the teacher can enrich his speech by using such an educational system (he presents to his students information which have easier to assimilate representations associated by the system).

Distance learning includes different ways of deployment and technologies to provide instruction (correspondence, audio, video, and computer). This implies a physical distance between the education's actors (student-teacher). The eLearning systems recognize this distance and they are trying to substitute it with a variety of strategies to encourage the interaction between the student and the teacher by offering new possibilities: the exchange of messages, documents or answers to required tasks.

An eLearning system (for distance training or virtual education) consists of a planned teaching-learning experience and it is organized by an institution which provides mediated materials in sequential and logic order to be treated by students in their own way, without forcing the participants to be present in a certain place at a certain time and to carry out a certain activity. Mediation is done in various ways, from the material on CD (possibly sent by mail) to technologies for transmitting content via the Internet.

For example, AeL (Advanced eLearning)[5] is an integrated system for teaching/learning and content management, which facilitates the work of the actors involved in designing and deploying the educational process: teachers, students,

[4] ELearning: http://en.wikipedia.org/wiki/E-Learning

[5] AeL: http://advancedelearning.com/index.php/articles/c322

evaluators, content developers. Although initially it was built for universities (especially for the form of distance learning), currently it is used in school education being extremely suitable for studying different languages, regions, levels of knowledge or types of organizations. The AeL platform, designed in multilayer system, represents: a standard client application web browser type and an application server based on Java platform.

One of the main characteristics of our eLearning system is related to the possibility to provide a two-way communication between the teacher and the learner with aim to reduce this distance between them. So, after the students log in they can see the electronic material for current courses and labs, can ask questions and can see the answers to their questions or to another questions, and the teacher can see these questions and can provide answers to them, all acting in real time.

3 User Modeling

Because of the continuous expansion of the Internet and the WWW's (World Wide Web), interconnected system of documents accessible through the Internet, and of the increasing number of computer users, the software systems must be increasingly more adaptable and that means that the systems must adapt depending on its user's interests, skills and experience (the users can be from a wide range of areas).

Although there were built graphical interfaces for computers more accessible for different types of users (as work field and interests, age and experience), there were not yet built good interfaces for each user.

Creating user models (User Modeling) is a research area which tries to build models of human behavior in an environment where human-computer interaction is specific. The purpose is not to imitate the human behavior, but to make the program to understand the user's desires, needs and expectations during the interaction. Moreover, the aim is that the system to be able to help the user to solve certain tasks (available/proposed by the program). The computerized representation of the user is called user model and the systems which create and use such models are called modeling systems.

Our modeling system combines ideas from adaptive hypermedia systems [2] and from recommendation systems [3, 7], and additionally we come with techniques from computational linguistics which allow us to extract sentiments from texts or to identify "*similar questions*". Thus, our system comparable with an adaptive hypermedia system, built the user's models starting from user's interests, desires and levels of knowledge and also it adapts the program's interface accordingly during the interaction with the corresponding actors. Similar with the recommendation systems our system is able to recommend to students which professor is most appropriate to a certain type of question or to recommend similar questions with available answers.

The features used by our system to obtain a user's model are:

a. *Knowledge* – for that we consider a basis accordingly with the student's current year (e.g., 2nd year), but we consider additional information, like grades at taught courses for a specific domain. For example, to analyze the

questions related to an advanced course we consider grades from basic courses followed in previous years.

b. *Interests* – represent the competencies that the user would like to acquire. We suppose that the interests of the students are related with the areas in which they ask questions. For example, if a student asks a question about a discipline called "Advanced Java Techniques" we suppose he wants to assimilate more about some specific Java techniques.
c. *The intention* – represents the user's aim during its interaction with the adaptive system. The student's intentions are: asking questions, finding responses, consulting the recommendations proposed by the system, filtering the existent questions. Through these options provided by the system, the student can find support to resolve homework from different areas or for the projects work preparation.
d. *Previous experience* – this is relevant for the system: work experience, the ability to speak a language. The student's experience is supposed to be in accordance with the year of study (e.g., A student in the third year has experience in courses from second year as opposed to a second year student who only studies these courses).
e. *Individual traits* – the components which define the user as an individual are: personality, cognitive traits, learning style and these are extracted through personality tests. In terms of individual features, the system focuses on each student's type of character and on identifying the appropriate scope of his work and they were extracted by two personality questionnaires.
f. *Work context* – it is approximated by elements such as the following: user's platform, location, physical environment, personal background.

We gave to users a list of areas about which to ask questions to a specialist and we considered that their interests are the areas that they choose to raise the question with the aim of finding out more information about it. The list contains the following subjects: current courses (*Software Engineering*, *Advanced Programming in Java* etc.) and general information courses (*Work License*, *Research Projects*, etc.).

The eLearning system collects explicitly information about the user on basis of two psychological tests. Thus, the system retrieved some personal traits of the student that couldn't be taken otherwise, namely, the type of character (*nervous, emotional, violent, passionate, sparkle, phlegmatic, amorphous or melancholy*) and the type of area in which they would like to work (*conventional or conformist, entrepreneurial or persuasive, investigative or intellectual*).

Additional to information's extracted explicitly from a user we used a software agent to extract information automatically. In this way, we have enriched the knowledge base about the user drawing some conclusions on the information entered explicitly (e.g., if a user explicitly specifies that he needs a response immediately and he receives it in less than 24 hours we may conclude that he's a happy user). Also, a professor can be informed by an agent about the student's grades from previous years, in order to help him to understand better why the student addressed some question.

4 Sentiment Extraction

The proposed system is based on the idea that in a text the words have no emotional charge (they describe facts and events), but they are emotionally charging according to the interpretation of each reader and each author's intention (in accordance with their interests) [1]. These interests are usually composed of personal needs, concepts that meet these needs, motivational factors, social and historical knowledge of facts, information circulated in media. These factors are called "knowledge base", which is generally composed of the general knowledge of words and their meanings, affective terms and emotion triggers.

An *emotion trigger* is a word or a concept in accordance with the interests of the user which lead to an emotional interpretation of the text's content. With these words, we built a database which enables us to classify and determine valency and feelings from text.

We will now present how we identify and how we classify the valences and the emotions presented in the texts written by students. To do this similar to [1], first we build incrementally a lexical database for Romanian language (which contains words that trigger emotions) to discover the opinions and emotions in the text at the word level (recognizing in it the positive, negative or neutral side of the sense). The second step is to assign valences and emotions to the terms from the database, and the third step is to identify the valency modifiers.

First step: In order to build a database with words that represent emotion triggers, we start from terms presented in "Maslow's pyramid" (it contains the hierarchy of human needs which are about 30 in English) and we translate them to Romanian. Because the number of terms is relatively small we disambiguate them using Romanian WordNet [8], and after that we associate with every term the set with synonyms accordingly with words sense. In this way we are sure that henceforth each new word will hold the meaning for which was added.

After that, we used again Romanian WordNet in order to add for every term all corresponding valid meanings and valid grammatical categories from Maslow's pyramid. For these new words we add hyponyms and the words which are in entailment relation.

Similar, we apply the same steps to the terms that are parts of the Max Neef's matrix [5], who believes that human needs are equally important, few, finite and classifiable.

Additionally, we consider that the terms related to exams or to dead-lines are also emotion triggers and we added to our database terms like: *punctaj* (En: *score*), *notă* (En: *note*), *termen limită* (En: *dead-line*), *examen* (En: *exam*), *parțial* (En: *partial exam*), etc.

The second step has as an aim to assign valences to the emotion trigger terms from the database built in step 1. For this the following rules presented in [1] were taken in account:

- The main trigger emotions and their hyponyms are given a positive value.
- The antonyms of the above terms are given a negative value.
- The term's valence is modified according to the modifiers (terms that deny, emphasize or diminish a valency) of which are accompanied.

At **third step** we define a set of valence modifiers (shifters) in Romanian starting from English modifiers from [1] in order to determine the changes in meaning of above emotion triggers. Additionally we add specific shifters accordingly with the courses followed by the students. The shifters can change a term's meaning radical, from positive to negative or vice versa, or can change a term's meaning and to make it neutral. We consider the following set of shifters that contains:

a. Negation words: *niciodată* (En: *never*), *nu* (En: *no*) that change the valance's sign.
b. A set of adjectives that enhance the meaning of a term: *mare* (En: *high*), *mai mult* (En: *more*), *mai bine* (En: *better*), *profund* (En: *intense*).
c. A set of adjectives that diminish a term's meaning: *mic* (En: *small*), *mai puțin* (En: *less*), *mai rău* (En: *worse*), *mai degrabă* (En: *rather*).
d. A set of modal verbs *a putea* (En: *can*), *a fi posibil* (En: *to be possible*), *a trebui* (En: *should*), *a vrea* (En: *to want*). They introduce the concept of uncertainty and possibility distinguishing between events that have occurred, could take place, are taking place or will take place in the future.
e. A set of adverbs that emphasizes the meaning of all content: *cu siguranță* (En: *definitely*), *sigur* (En: *certainly*), *cert* (En: *sure*), *în definitiv* (En: *eventually*).
f. A set of adverbs that change valence and diminish the entire context's emotion: *posibil* (En: *possible*), *probabil* (En: *probable*).
g. The terms which add a note of uncertainty to the context: *abia* (En: *hardly*). These terms may add uncertainty to the positive valence of context, even if in the text does not exist other negative terms.
h. Connectors: *chiar dacă* (En: *even if*), *deși* (En: *although*), *dar* (En: *but*), *din contră* (En: *the contrary*) can enter information, but can influence the text they belong.

Negations are used to switch to the opposite meaning of a term while intensifiers and modifiers have role to increase or to decrease the valency degree of a term.

The main observation to be mentioned here is the following: for a valency modifier to fulfill its purpose (to alter the term's valence), in the text must be expressed an attitude (who's understood to be modified).

5 The System

From the beginning we established the site's purpose: to build a consistent database with models for students, after investigation steps, in order to identify and to interpret sentiments from student's questions. To achieve this objective we built two interfaces: one for second year students from our faculty (the Faculty of Computer Science, the Alexandru Ioan Cuza University of Iasi) and one for their professors. The system's architecture is presented in Figure 1.

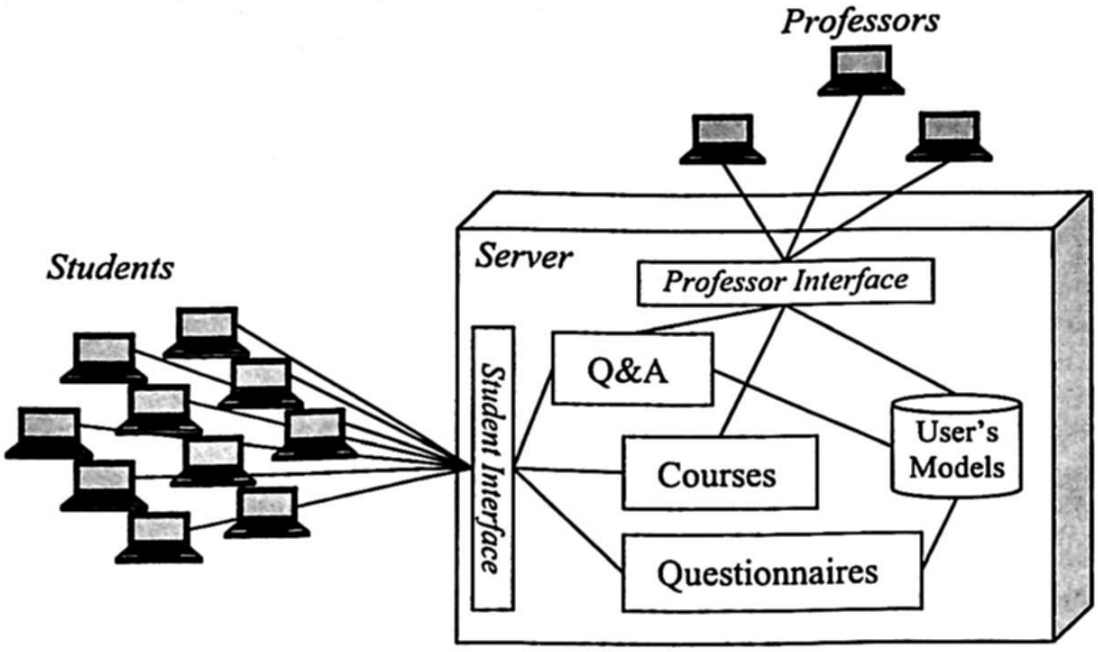

Fig. 1. The system's architecture

Student's Interface: after registering, every student receives an account that allows him to communicate with his professors. A student has access to the following components:

a. *Courses page* – here a student can find the available courses and can assist to a preferred course. For every course there are electronic materials, presentations, practical or theoretical exercises and useful links.
b. *Question&Answering page* – allows and facilitates discussions between students and professors. Here, the students can ask questions related to current courses or general questions (regarding the work license etc.). Every question has additional information about priority (*normal*, *urgent*, *trivial*) and about the student's motivation (*to solve problems*, *to have more knowledge*, etc.). Moreover, the students may request a meeting with the teacher when they believe that a discussion face to face would help them more.
c. *Questionnaires* – that help our programs to build the user's model. Here we consider two types of questionnaires: first is for the character type *(nervous, emotional, violent, passionate, sparkle, phlegmatic, amorphous or melancholy)* and the second one is related to the area where they would like to work *(conventional or conformist, enterprising and persuasive, investigative or intellectual)*. The first test is composed of 12 questions and the second test of 25 questions and it was constructed following the Holland test[6] model.

The Professor's Interface: after registering, every professor receives an account that allows him to communicate with his students. A professor has access to the following components:

[6] Holland test model: http://www.hollandcodes.com/my_career_profile.html

a. *Courses page* – in order to add new courses and new materials for them. Also a professor can see a student's comments related to courses, the number of students that attend the courses and the number of students that resolve exercises and their proposed solutions.
b. *Question&Answering page* – the professors can answer to questions related to their courses, but they may also respond to general questions or to questions related to other courses. The order of answers depends by the priority of the questions and the answer's content depends by a student's motivation (a simple answer if the student wants to solve an exercise or a detailed answer if the student wants to use some techniques in order to implement complex projects, etc.). In addition, the teachers can arrange meetings with students who have requested it explicitly.
c. *User's Models* – this page can be accessed from Question&Answering page with aim to understand better the question and in order to identify which is the desired answer.

Both the teachers and the students have the opportunity to filter the questions according to certain criteria. Therefore the teachers get a list of questions:

- Sorted in ascending/descending order by date the student addressed the question.
- Sorted according to the priority specified by the student.
- Addressed on a particular subject by all students.

Students get a list of their own questions:

- Sorted by date when the professor offered the response.
- Sorted by priority specified by them.

On Question-Answering page, we add a specific module that analyzes the question and shows to student the similar questions with their answers. This module processes the question and identifies the *question type* (which can be *Definition*, *Factoid* or *List*), the *answer type* (which can be *a date, a number, a link address, an organization, other, etc.*) and the *keywords* (which are the most relevant words from the question) similar to [4]. Having these values we used two methods to search *similar questions* for a given question:

1. The first method considers previous questions and concludes that they are similar if they have the same values for question type and answer type and if they have common keywords. For example we consider the following questions: the initial question is: *Care este termenul limită pentru alegerea temei proiectului?* (En: *What is the deadline for choosing the theme of the project?*) and the current question is: *Până când trebuie ales proiectul?* (En: *Until when should be chosen the project?*). We can see that both questions are Factoid questions and expect an answer of type Date. Also, we have one common keyword: *proiect* (En: *project*). For these reasons we consider that the questions are similar and we offer to that student for analysis the answer to the initial question.
2. The second method is applied when the question or the answer types are different, but the questions are from the same period and are related to the same course. Again, in this case we offer for analysis the answer of the initial question.

6 Statistics

During a semester we offered to our second year students the possibility to ask the teachers and to find out new things in certain areas. 132 students have created accounts, representing over 50% from the total number of students and 112 put at least one question. The total number of questions was 305, meaning that on average each student asked about 3 questions. The maximum number of questions asked by a student was 11. In Table 1 we can see the distribution of questions for professors.

Table 1. The distribution of questions for professors

Professor ID	Questions number	Percent
1	7	2.29 %
2	20	6.56 %
3	124	40.66 %
4	83	27.21 %
5	21	6.89 %
6	41	13.44 %
7	9	2.95 %

We studied the questions and we noticed that the teachers who received more questions are professors who perform additional research activities with students or professors who have complex problems at laboratory practical activities. We discussed with the professors who received a large number of questions (IDs 3, 4 and 6) about the advantages and disadvantages of using the system that makes the user's model.

The system has seven teachers involved: two teachers did not used the system at all (IDs 1 and 7), two of them used the system very little because they didn't believe in the effectiveness of the system (IDs 2 and 5) and three teachers used the system long enough (IDs 3, 4, 6).

The third professor believes that the system has improved his communication with the students and would also use it in the future to give students more opportunities to keep in touch. The same professor is the one who regularly used the models created by the system for users before providing answers for questions. The professor said that about 50% of the models were useful (he was satisfied with the way they were created and their contents) and about 50% of the models were unnecessary (the created models were incomplete or contained no useful information to answer the questions).

The system has 112 students involved, but we have done the research studying 40 of them (each of these students put more than 5 questions). Our models for users are built iteratively from question to question, using additional information obtained on questionnaires basis. When professors receive a question, they receive additionally the user's model built from previous questions and an analysis of the current question. Let's see few examples.

For this question "*Aş putea să fac un proiect cu informaţii despre hoteluri?*" (En: *Could I make a project with information about hotels?*) the student specified that the question is urgent and that he needs mandatory an answer. Because the student used

in his question "*aş putea*" (En: *could*) our application identifies the uncertainty of the student who does not know the possible alternatives. This question has a higher positive valence and it is different from the previous types of his questions that have a lower positive valence or even are neutral. In this kind of situation our application mentions that something happened and that the student needs more information. After some verification we identified that our supposition was correct: the student missed one week because he was at a student contest.

From 305 questions our system marked 114 questions that have emotion triggers terms and calculated the valences for them. In 9 cases the positive valences were higher than the rest of the marked questions. One of these questions is "*Dacă pentru realizarea aplicaţiei pare mai uşor să încalc un design pattern, decât să îl respect, trebuie totuşi să urmez patternul respectiv sau e în regulă să fac aplicaţia cum consider de cuviinţă, atât timp cât rezultatul final funcţionează?*" (En: *If in order to implement an application seems easier not to respect a design pattern, instead to respect it, must I still follow that pattern or is it ok to make the application how I think, as long as at the end the application works fine?*). For this question we identified the following emotion triggers: *pare* (En: *seems*), *mai uşor* (En: *easier*), *decât* (En: *instead*), *trebuie* (En: *must*), *totuşi* (En: *still*) and in this case we obtained the highest value for valence. Interesting is the fact that this question was the first question asked by that student and the rest of his questions, even if they were without emotions triggers, were influenced by this question.

After several discussions with professors, we decided to offer at request for a new question the user's model for the student who asks the current question, obtained from all his questions. Additionally the system remarks differences between the valence values obtained for the current question and for previous questions. Also, the professor receives the user's comportment type and the user's future goals.

7 Conclusions

Adaptive hypermedia is the answer came to help the user who is "lost in space" because there are too many links from which to choose or because he doesn't know how to find the shortest way to achieve the personal goal. In the application we created we tried to help the user using natural language techniques.

Thus, we helped a student by suggesting him the answers to similar questions offered by a teacher and we helped a teacher to understand better the question giving him the opportunity to access a user model which we created for that student. In the first case we used the question-answering techniques and in the second case we combined the psychological profiles of students with profiles built on feelings drawn from the questions submitted by students.

In recent years there have been various methods to identify feelings in a text: the explicit request of a transmitter's opinion, identifying the feelings directly related to different areas of interest. In this paper, the emphasis was placed on the role of emotions and on triggers terms and more than that those feelings were identified in the text by identifying the positive, negative or neutral aspects of words.

The assessment regarding the opinion of the students about the created system was not realized but we want in the future to create a questionnaire that would allow this. Also we want to talk with the students who used the system more because we want to improve the components dedicated to them according to their preferences.

References

1. Balahur, A., Montoyo, A.: Applying a culture dependent emotion triggers database for text valence and emotion classification. In journal Procesamiento del Lenguaje Natural, ISSN 1135-5948, N°. 40. Pp. 107-114. (2008)
2. Brusilovsky, P., Millan, E.: User Models for Adaptive Hypermedia and Adaptive Educational Systems. The Adaptive Web Journal. Pp. 3-53. (2007)
3. Brut, M.: Ontology-Based Modeling and Recommendation Techniques for Adaptive Hypermedia Systems. (Ph.D. Thesis) Technical Report 09-04. "Al. I. Cuza" University. ISSN 1224-9327. 155 pages. Iasi, Romania. (2009)
4. Iftene, A., Trandabăț, D., Pistol, I., Moruz, A., Husarciuc, M., Cristea, D.: UAIC Participation at QA@CLEF2008. In Evaluating Systems for Multilingual and Multimodal Information Access. Lecture Notes in Computer Science. Vol. 5706/2009. Pp. 448-451. (2009)
5. Manfred, A.: Max-Neef with Antonio Elizalde, Martin Hopenhayn. Human scale development: conception, application and further reflections. New York: Apex. Chapter 2. "Development and Human Needs". Pp. 18. (1991)
6. Maslow, A.H.: A Theory of Human Motivation, Psychological Review 50(4). 370-96. (1943)
7. Tran, T., Cimiano, P., and Ankolekar, A.: A Rule-Based Adaption Model for Ontology-Based Personalization. Springer, Volume 93. Pp. 117-135. (2008)
8. Tufiș, D., Ion, R., Ide, N.: Word Sense Disambiguation as a Wordnets' Validation Method in Balkanet. In LREC-2004: Fourth International Conference on Language Resources and Evaluation, Proceedings, Lisbon, Portugal, 26-28 May 2004. 1071-1074. (2004)

Predicting the Difficulty of Multiple-Choice Close Questions for Computer-Adaptive Testing

Ayako Hoshino[1]*
Hiroshi Nakagawa[2]
[1]NEC Common Platform Software Research Laboratories
[2]Unviersity of Tokyo
a-hoshino@cj.jp.nec.com, n3@dl.itcu-tokyo.ac.jp

Abstract

Multiple-choice, fill-in-the-blank questions are widely used to assess language learners' knowledge of grammar and vocabulary. The questions are often used in CAT (Computer-Adaptive Testing) systems, which are commonly based on IRT (Item Response Theory). The drawback of a simple application of IRT is that it requires training data which are not available in many real world situations. In this work, we explore a machine learning approach to predict the difficulty of a question from automatically extractable features. With the use of the SVM (Support Vector Machine) learning algorithm and 27 features, we achieve over 70% accuracy in a two-way classification task. The trained classifier is applied to a CAT system with a group of postgraduate ESL (English as a Second Language) students. The results show that the predicted values are more indicative of the testees performance than a baseline index (sentence length) alone.

1 Introduction

Multiple-Choice (MC) questions are commonly used in many standardized tests, due to its ease of collecting and marking the answers. Multiple-Choice Fill-In-the-Blank[1] (MC-FIB) questions, especially, have proven their effectiveness in testing language learners' knowledge and usage of a certain grammar rule or a vocabulary item. The TOEIC[2] test, for example, has questions in this format. Below is an example of a multiple-choice cloze question:

There is a [ ] on the tree.
1) bird 2) birds 3) orange 4) oranges

The sentence is called the *stem*, in which the blank is shown as the brackets. The alternatives consist of one right answer and the distractors.

*This work has been done as a part of Ph.D. studies at University of Tokyo.

[1]For brevity, we use *cloze* instead of fill-in-the-blank.

[2]Test of English for International Communication, run by ETS (Educational Testing Services), U.S.

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 279-292

Received 04/10/09
Accepted 16/01/10
Final version 10/03/10

CAT is a technology that are motivated to offer a better assessment by being adaptive to the testee, where the computer administers subsequent questions depending on the precedent performance of the testee. IRT provides the theoretical background on most of the current CAT systems, in which a testee's ability (or *latent trait*) and the difficulty of a question are described in values in a common unit called *logit*. In IRT, the difference in the ability and difficulty is projected to the probability of the users' getting the right answer, using a sigmoid function. For a fuller introduction to IRT, the readers are guided to Baker et al. [1].

IRT is a well-researched area, where many positive results have been reported. For example, Urry reports that an IRT-based CAT system achieves sufficient precision with 20 questions, whereas pen-and-paper testing requires 100 questions [2]. However, less attention has been focused on the cost of adapting the model, which is, as commonly practiced, the cost of conducting the pre-test results on the comparable group of testees. In cases where pre-testing is not possible, all questions are assumed to be of equal difficulty at the onset, then the difficulty of the questions is updated as the users' responses accumulate [1].

This research is motivated to combine CAT with recently emerging AQG technology (Automatic Question Generation, explained in the following section), which will render possible a novel assessment system that adaptively administers questions from the automatically generated question set. However, one obstacle is that the above-mentioned problem of adapting an IRT model to the testees' level, whose cost will be higher as the available number of questions increases. Existing methods of IRT model adaptation will be inpractical when unlimited number of newly-generated questions are added to the question pool.

In this situation, it is vital to have a means of automatically calculating a rough prediction of difficulty, or inferring the difficulty of a question from the performance of the targeted group on similar questions. The use of supervised machine learning would be worth exploring, which will also be an attempt to have the computer gain a general notion of the difficulty of MC-FIB questions.

The rest of the paper is organized as follows: We review relevant work on difficulty prediction in Section 2. In Section 3, the proposed method is presented with evaluation results on a closed data set. In section 4, the trained classifier is tested in a subject group experiment. Section 5 concludes the paper.

2 Related Work

There is only limited literature in the field of computational linguistics on MC-FIB questions for language proficiency testing. Among the attempts in generating MC questions from an input text [3] [4] [5] [6] [7]. One of the AQG studies provide a method to compute the complexity of reading comprehension questions [8][3] to be used in a CAT system. Their measure is defined as a weighted sum of complexity values on the stem sentence, the paragraph, the answer sentence and so forth.

[3] In their study, the format of questions was neither MC nor FIB, thus the answer is composed by the testee.

In fact, the complexity of a sentence alone has been studied for decades in computational linguistics, and many indices have been proposed. Segler, in his work on extracting example sentences for language learners, compares traditionally proposed indices [9]. The indices include sentence length (number of words in the sentence), parse-tree depth (maximum number of levels from the root to a word), and the combination of such factors. Segler's comparison reveals that it's very hard to beat the sentence length, which is the simplest measure.

The complexity of the stem sentence affects the difficulty of the above-presented question. But other factors, such as similarities between the right answer and the distractors would surely influence the difficulty of MC-FIB, and thus should be taken into account.

The Flesch-Kincaid Index is a readability measure for a passage, which is widely used among educators. The index is defined as follows:

$$R_{FRE} = 206.8 - 1.05X - 84.6Y$$

where X is the average number of syllables in a word and Y is the average number of words in a sentence. The index can be applied to a sentence with Y fixed to one. This version of Flesch/Kincaid score is composed so the value is interpreted as the grade in an American elementary school.

Some improvements of the readability measures have been proposed [10]. Miyazaki et al. proposed an individualized measure for reading text extraction [11].

Evaluation of the existing indices are done often manually with add-hock parameters. In this study, a supervised machine learning technique is used to tune the parameters in combining feature values.

3 Difficulty Prediction

In this study, we use the technique of supervised machine learning for the task of difficulty prediction. We first explore the learning algorithms, and train the best performing classifier using the question data that are annotated with the correct response rate. Then, with a simple binary search-like method based on the predicted difficulty values, we build a CAT system and have it tried out by human subjects.

As this is one of the earliest attempts in applying machine learning methods to such a task, we set out with a simple binary classification. We did not employ regression, as some of the readers may wonder, which outputs numerical values. The reason was that 1) it is expected to be unworthy to predict the correct response rate that is observed from subject groups, since such observation usually contains what is called *measurement error* in the literature of psychometrics. We train the classifier with the labels "easy" or "difficult," letting the computer to try to grasp a rough notion of difficulty.

3.1 Training Data

The training data set is obtained from a series of TOEIC preparation books (a total of 702 questions from Part 5, which are MC-FIBs.) Each question is annotated with the correct response rates, ranging from 0.0 to 98.5. The figures are based on the tests in

a TOEIC preparation school in Japan and reportedly based on the results of about 300 testees. Table 6 (in Appendix) shows samples from the training data.

All questions consist of a stem sentence of 20-30 words and four alternatives. Seemingly, all questions are intended to be of the same difficulty, rather than being increasingly difficult according to the question number. We have labeled the top 305 easiest questions as "easy" and the top 305 difficult questions as "difficult," based on the correct response rates, leaving out 8% around the average value [4].

3.2 Features

On deciding the feature set, we take a similar approach to Kunichika et al., who designed the factors of difficulty depending on the complexity of the question and of the answer. We assume the difficulty of a question to be composed of 1) the difficulty of the stem sentence, 2) the difficulty of the correct answer, and 3) the similarity between the correct answer and the distractors. As MC-FIB questions are often criticized for being possible to obtain the right answer just by reading a few words before and after the blanks, we have also added as a feature as 4) the tri-gram including the correct answer or a distractor. Each feature, with its notation used in this paper, is explained as follows:

1) Sentence features
The sentence features consist of **sentencelength**, which is the number of words in the original sentence, **maxdepth**, which is defined as the *depth*, or number of brackets/levels from the root to the deepest word in a parse result [5], and **avr_wlen**, which is the average number of characters in a word.

2) Answer features
The answer features provide information on the right answer, consisting of **blanklength**, which is the number of words in the correct answer, and an array of binary features on the POS (Part Of Speech) of the right answer (**pos_V**, **pos_N**, **pos_PREP**, **pos_ADV**, and **pos_ADJ**), which indicate inclusion of the part of speech in the right answer. For example, **pos_V** is true if any form of a verb is included in the correct alternative.

3) Distractor similarity features
The distractor similarity features are obtained from the analysis using the technique of modified edit distance, which have been used to extract a lowest-cost conversion path from one string to another. In our version of edit distance, we have applied the algorithm on a word basis, as opposed to the character based application as seen in spelling-error correction. While three kinds of operations, *insert*, *delete*, and *change*, are used in a standard edit distance, we have additionally defined three operations: *change_inflection*, *change_lexical_item*, and *change_suffix*. *Change_inflection* is an operation where a word is substituted by the same vocabulary item, but in a different inflectional form. *Change_lexical_item* is a substitution of a word with the same vocabulary item in the same inflectional form. *Change_suffix* is a substitution of a word

[4] This 8% was decided in a cross validation; we took a breaking point that maximizes the accuracy.
[5] We used the Charniak parser `http://www.cs.brown.edu/~ec/`.

Table 1: Results with different learning algorithms

Algorithm	Accuracy
SVM	**62.7960%**
SMO	58.1481%
Logistic	57.4074%
VotedPerceptron	57.0370%
IB5	57.0370%
NaiveBayes	56.6667%
SimpleLogistic	56.4198%
IB10	55.5556%
MultilayerPerceptron	53.9506%
J48	53.7037%
IB3	52.8395%
RBF Network	52.2222%
IB1	51.8519%

with another word with the same stem, which can be of a different part of speech. We set the cost of operation so the *change_inflection* (cost: 2), *change_lexical_item* (3), *change_suffix* (4), and standard *change* (5) are preferred in this order. The cost of *insert* and *delete* is set to 3, so the combination of deletion and insertion is never used when one change yelds the same result.

The features derived from the analysis are: **pathlength** is defined as the number of operations, and an array of binary features (**include_insert, include_delete, include_changeinflection, include_changelexicalitem, include_changesuffix, include_change**). For example, **include_insert** is true if any of the conversions from the right answer to the three distractors include an operation *insert*.

4) Tri-gram features

The tri-gram features are obtained from a search engine's hit score, assuming that the figure reflects the familiarity of a given tri-gram. Tri-gram features on the correct answer are **hit_correct, hit_pre2_corr, hit_pre2_corr, hit_pre_corr_post**, and **hit_corr_post2**, where **hit_correct** is the google hit score of the correct answer, **hit_pre2_corr** is the same score of the correct answer with the previous two words, **hit_pre_corr_post** is the hit score of the correct answer with the previous word and the following word, and **hit_corr_post2** is the hit score of the correct answer with the following two words. The same set of features is defined for the distractors, where **hit_distractor** is the average of the google hit score of the three distractors. All queries are posted in parenthesis.

All features are turned into numerical values, normalized[6] before being fed to the learner.

3.3 Learning Algorithms

We conduct the 10-fold cross validation to compare the performance of different learning algorithms from weka machine learning toolkit [12] and SVM_{light} applied to this task. Table 1 shows the accuracy of each learning algorithm.

To our disappointment, many of the learning algorithms performed no different from sheer randomness (50%). The best accuracy is achieved by SVM_{light}, noted as

[6]We use a normalization filter in *weka* package (`http://www.cs.waikato.ac.nz/~ml/weka/`). We have also tried standardization, but did not obtain better performances than normalization.

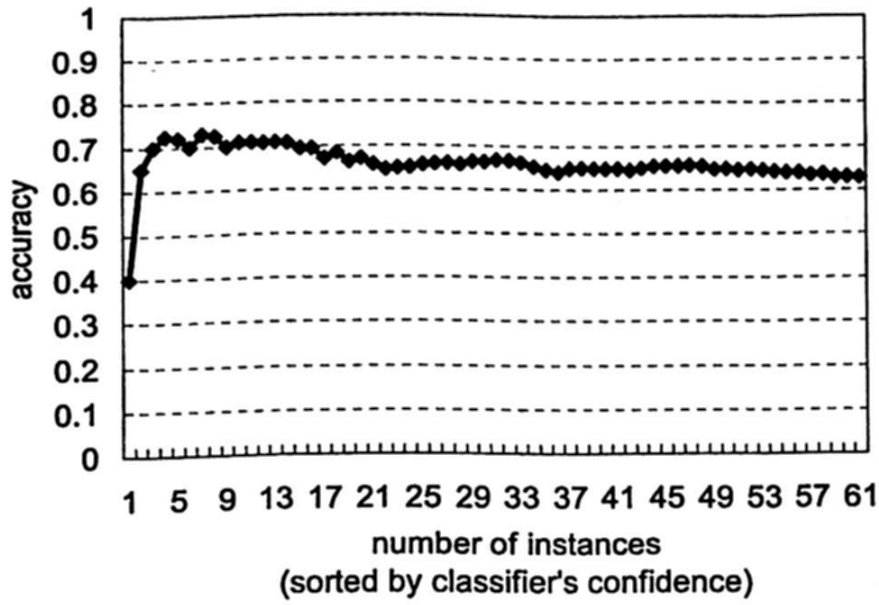

Figure 1: Accuracy on top n instances

SVM in Table 1, outperforming complex algorithms such as Voted Perceptron. Although, some of the classifiers have unexplored parameters (where we have used default values) that could change the resulting accuracy, we concluded that SVM is the most suited algorithm for this task. The fact that SMO (Sequential Minimal Optimization), which is also a version of SVM, performs second best supports the hypothesis that the high-dimensional maximum margin approach is effective for this problem.

One can observe that the parameter k optimizes IBk, or k-nearest neighbor algorithm, at around 5, though the gain from IB1 remains about 5%.

3.4 Top-N Accuracy

Using SVM_{light}, we conducted another cross validation with top N evaluation. Valuing more precision than recall, top N evaluation is an evaluation where the test instances are sorted by the confidence [7] of the classifier and the accuracy is calculated on the top N confidently classified instances. The figure in top N accuracy can be interpreted as the performance of the classifier when it is allowed to skip less confident instances. The result of top-N evaluation is shown in Figure 1.

The line indicates the mean accuracy of 10 split, averaged over the results of 1,000 runs. The x-axis shows the number of instances the classifier has labeled, and of which the classifier is most confident. The mean accuracy draws a sharp curve at first, achieving 72% with about 10 instances at the top. The accuracy is kept above 70%, up until the top 18 instances. Then, it slopes down marking 65% at top 50 instances and gradually slopes down reaching 0.629% in labeling all instances. The overall standard deviation was about 0.005, which indicates the classifier's stability. On the condition

[7] The confidence value in the context of the SVM algorithm is the distance to the separating hyperplane. We use this index as *difficulty value*, which ranges from negative to positive values, and in our case, smaller value signifies more difficult; if the classifier is more confident of a question's being difficult, it is a more difficult question.

Table 2: Accuracy gain of each feature (Left: those with positive contribution, Right: those with zero or negative contribution)

Feature f	w/o f	Gain
all features	**0.70769**	-
pos_N	0.65385	0.05385
pos_V	0.66154	0.04615
sentencelength	0.66923	0.03846
hit_pre2_dist	0.66923	0.03846
include_change	0.67692	0.03077
maxdepth	0.68462	0.02308
avr_wlen	0.68462	0.02308
hit_pre2_answer	0.69231	0.01538
hit_pre_dist_post	0.69231	0.01538
include_changesuffix	0.70000	0.00769
hit_dist_post2	0.70000	0.00769
hit_pre_answer_post	0.70000	0.00769

include_insert	0.70769	0.00000
answer_wlen	0.70769	0.00000
pos_ADV	0.70769	0.00000
blanklength	0.70769	0.00000
include_delete	0.71538	-0.00769
include_change _lexicalitem	0.71538	-0.00769
include_change _inflection	0.72308	-0.01538
pos_PREP	0.72308	-0.01538
hit_dist_count	0.72308	-0.01538
pathlength	0.73077	-0.02308
pos_ADJ	0.73077	-0.02308
hit_answer_post2	0.73077	-0.02308

that the predictor works similarly well on the automatically generated questions, the predictor could label the question with over 70% accuracy skipping three out of four instances.[8]

Note that binary classification can be extended to comparing and ranking two or more instances. SVM algorithm can be used for comparing two or more candidate instances, which is the way a classifier will work in an actual CAT system.

3.5 Feature Analysis

We investigate which of the features have contributed to the accuracy of the classifier, by removing one feature at a time. The difference from the accuracy with all features signifies the gain of accuracy caused by the feature. The experimental settings was the same as the above cross validation. Table 2 shows the accuracy and the performance gain with the top 15 instances, where the performance is kept above 70% with the largest number of labeled instances.

The most contributing feature was **pos_N**, with the gain of 5% of accuracy. **Pos_V** provides the next largest gain. These two features exceed the contribution of the sentence length, which is assumed to be an undoubted predictor for sentence difficulty. Then, information on the hits scores and the operations in the paths follows, with **include_change** and **include_changesuffix** providing larger contributions. The four features (**include_insert**, **blanklength**, **pos_AD** and **hit_answer**), however, do not affect the accuracy at the point of top 15. Several features, such as **hit_answer_post2** and **pos_ADJ**, exhibit negative contributions, though those features contribute by large margins as more instances are labeled. **Avr_wlen** provides a larger contribution as more instances are counted as top n.

[8]Note that skipping unconfident instances does not cause a problem in our AQG+CAT system, since unlimited number of automatically-generated questions are available.

To summarize the results of the above experiments: 1) the overall performance is higher with the instances with higher confidence values than all instances, 2) path features contribute to the accuracy, 3) **include_change** and **include_changesuffix** contribute more than the other path-related features, while features such as **include_insert** don't provide much information at the point of $n = 15$, and 4) the POS information on the answer phrase helps the accuracy boost, depending on which POS to look at. Information on the verb or noun inclusion in the answer phrase significantly contributes to the accuracy. However, adverb, preposition, and adjective do not result in positive accuracy gains. 5) Some of the features do not look effective with fewer number of confident instances, although many of them prove effective with the larger number of instances.

The results of cross validation with different features removed also provide an analysis on the group of testees. In the case of our data, the feature contribution reflects the tendencies of the testees who go to the TOEIC preparation school. It could also be possible to assume that the group of testees is a good sample of adult Japanese learners of English. Since the method of cross validation allows us to repeat the experiments with different features, the researcher of SLA (Second Language Acquisition) can perform analysis with their own devised features, without the need to collect the data with a carefully designed subject experiment. Also, provided with the sufficient amount of the data with a given learner, the analysis can provide a personal diagnosis of their tendencies.

3.6 Human Performance

The difference among the difficulties of the questions taken from the same series of books was quite subtle. To see the difficulty of this task for human judges, we asked two Japanese assistant professors in computer science to perform the binary classification. They were presented with a mixed set of 40 "difficult" questions and 40 "easy" questions and guessed the label of each instance. The accuracy of the two human judges was 70% and 72.5%. Considering the fact that the performance of human subjects is normally deemed to be the upper limit of an NLP task, this not-too-high performance of SVM classifier makes sense. The subjects pointed the type of question was a clue they used to decide the labels; the grammar questions tended to seem easy, while vocabulary questions were more difficult.

4 Subject Experiment

In order to see the efficacy of a trained difficulty predictor in the context of AQG+CAT testing, we conducted a subject experiment. The questions we use in the experiment are automatically generated from online news articles, which are administered by a simple algorithm based on the predicted difficulty values.

The entire evaluation was conducted through the Internet. The subjects were called for and volunteered through the department's email list. The participants were instructed by email, tried out the CAT system through their Internet browser, and answered the post-task questionnaire by e-mail.

Twenty students responded to our call. They are master's and doctoral students majoring in information studies (with either literature or science background.) Their first languages are Japanese (12) and Chinese (6) and other languages (2).

4.1 Automatic Question Generation

With our in-house AQG system, we generated grammar and vocabulary questions on articles from several online news websites: NHK (Japan), BBC (U.K.), and DongA (Korea). The method of AQG we employed was Coniam's frequency-based method [13] for vocabulary distractors and hand-written patterns for grammar distractors. Table 7 (in Appendix) shows samples from the questions used in the experiment, along with their predicted difficulty value (first column) and the correct response rates (second column). In this subject experiment, a set of automatically-generated questions were labeled with a classifier trained in an abovementioned method, then administered to the subjects.[9] For more information on our AQG method, see [14].

4.2 Administration algorithm

Assessing a testee with a CAT system is done in a similar way as finding their position on a number line. The system starts with the questions of a mean difficulty, then it jumps a pointer (representing a participant's position) to go up or down depending on the participant's response. The width of the jump is reduced as more questions are administered, following an inverse logarithmic function as defined below:

$$C(top - bottom)/log(n + 1)$$

where *top* is the maximum and *bottom* is the minimum difficulty value in the question pool, which, at the beginning of evaluation, contained 3,000 automatically-generated questions. The value n is the number of questions attempted so far. The constant C is set by the simulation experiments. In this experiment, our system excludes the sentences that have previously been exposed to the participant.

4.3 Experiment results

We have conducted a three-session experiment, where the participants took part in two or all three sessions. The number of participants were 12 in the first, 15 in the second, and 17 in the third session. Fifty questions were administered at each session, where two sessions were with random administration and one session was with an adaptive administration. The basic information of the test results is summarized in Table 3.

[9] When applying the SVM_{light}'s classifier trained with the aforementioned data, the resulting values of the test data are extremely skewed. In fact, at most of the time, the same value is gained for all test data. This could be attributed to the difference between the training data and the test data. For example, the sentences from the news articles tend to be longer than the ones in TOEIC MC-FIB questions. Thus, the feature values (e.g., **sentencelength**) of the test data range outside the ones of the training data, resulting all instances being more difficult. We have re-run the training process with the option of preference ranking, with input training data (ranking) being all combinations of the "difficult" and "easy" instances. (About the option *preference ranking*, see the website of SVM_{light} `http://svmlight.joachims.org/`) With this setting, the resulting difficulty value with the test data distributed much like a normal distribution.

Table 3: Summary of the test results

	First session	Second session	Third session
Average correct response rate (stdev.) highest/lowest	0.785 (0.097) 0.980 / 0.627	0.741 (0.075) 0.860 / 0.600	0.758 (0.079) 0.920 / 0.660
Average total time longest/shortest	0:28:33 1:10:53 / 0:15:32	0:30:45 0:54:58 / 0:12:41	0:31:13 0:49:51 / 0:11:39

Table 4: Average of the two indices in two groups based on the observed difficulty

	sentence length		predicted difficulty	
average	difficult: 26.14	easy: 24.91	difficult: -1.626	easy: -1.512
variance	difficult: 72.09	easy: 115.89	difficult: 0.326	easy: 0.404
p value	0.6425		0.2183	

Table 5: Correct response rate by part of session

part	1-10	11-20	21-30	31-40	41-50	variance
random1	0.753	0.846	0.741	0.792	0.725	0.00188
random2	0.716	0.799	0.724	0.758	0.696	0.00131
adaptive	0.729	0.724	0.727	0.790	0.788	0.00094

The questions that have been administered to the participants were automatically-generated, hence were not always the errorless ones. Still, the correct response rate of the participants were quite high with 75% on average with the highest being 86-98%.

4.4 Information gain by difficulty prediction

There were 103 questions that were solved by more than three participants. At a first look, disappointingly, there was no significant correlation between predicted difficulty values and the correct response rates. We further took a look at the distribution of the correct response rates, and sampled the "difficult" questions and "easy" questions. We call them *observed difficulty* as opposed to predicted difficulty. There were 44 questions that were answered correctly by all participants who had been administered them. We labeled those questions as "observed easy," and labeled the questions whose correct response rate was 0.6 or below as "observed difficult." Table 4 compares the two indices 1) sentence length (a baseline), and 2) the predicted difficulty value, in their relation to the observed difficulty.

The results show that both of the indices differ in the two groups as expected; the average sentence length is larger, and the predicted difficulty value is smaller, in the observed "difficult" group. The p-value is smaller on the predicted difficulty, which means that the predicted difficulty value differentiates the two groups better than the sentence length.

A weak level of significance ($p = 0.2$) is observed on the predicted difficulty, despite

of the difference of the two sets of questions, as well as a rather diverse subject group. The predicted difficulty was calculated on the test results of an English school, whose students are mostly Japanese learners. On the other hand, the participants we gathered included many international students whose first language was different from the others. Also, the difficulty must have reflected the difference in nature of the professionally written TOEIC preparation questions and the automatically-generated questions. The former is free of context and generally very well-written, while the latter tends to be context-dependent, and although it has 10 different patterns, still it gives an impression of being pattern-generated. These differences will be incorporated to further improve the current system. For example, we can incorporate the observed data into the training data to better tune the difficulty prediction.

4.5 Transition of the correct response rate

Finally, we took a look at transitional changes in correct response rate in the three sessions to see the system's adaptivity to the human users. We have split a session into five parts by the order of administration. Table 5 shows the correct response rate calculated on each part along with their variances.

First, it is observed that the correct response rate is more stable in the adaptively administered session. It is generally a good sign for a test, since stability of the correct response rate can be attributed to the system's administrating questions of similar difficulty, rather than moving to and from the extremities. Second, adaptive session was the only one where the performance of the participant rose in the last half of the session[10], which was unexpected, since we were hoping that the correct response rate should fall as the CAT system administeres more suited and thus challenging questions to the user.

The rise of the correct response rate can be attributed to the users' habituation to the patterns, since the adaptive session was the third session for all participants. Also, it is known that a CAT system needs fewer questions than conventional pen-and-paper tests to reach the true values with minimized errors. As Urry reports, only about 20 questions are necessary in English grammar and vocabulary tests for adult native speakers. In our data, the adaptive session was the only one where the correct response rate did not rise on the second split. We speculate that the adaptive administration actually chose more difficult questions to the response to the high correct response rates of the users, achieving the near value after 11 to 20 questions, and then drifted away to the easier questions.

5 Conclusions

We have investigated an application of the machine learning techniques to the problem of difficulty prediction for a CAT system. The SVM classifier shows a performance on par with human judges, and the predicted values show some evidence of efficacy, showing more information gain than sentence length index alone, and stable correct response rates than random administration. Future direction includes more investigation

[10] Whose difference from the other sessions was significant in t-test with ($0.5<p<1.0$)

with the features, such as the use of syllable numbers and other difficulty measures for words. The problem of the difference of subject groups would be alleviated by re-training and updating the classifier as the data from the targeted group is obtained. Combining this prediction with standard IRT procedure also is an interesting avenue towards a more effective assessment system.

References

1. Baker, F.B., Kim, S.H.: Item Response Theory: Parameter Estimation Techniques, Second Edition (Statistics, a Series of Textbooks and Monographs). Marcel Dekker, Inc., New York, USA (July 2004)
2. Urry, V.W.: Tailored testing: A successful application of latent trait theory. Journal of Educational Measurement 14(2) (1977) 181–196
3. Sumita, E., Sugaya, F., Yamamoto, S.: Measuring non-native speakers' proficiency of english by using a test with automatically-generated fill-in-the-blank questions. In: Proceedings of the Second Workshop on Building Educational Applications Using Natural Language Processing, Ann Arbor, Michigan, U.S., Association for Computational Linguistics (June 2005) 61–68
4. Liu, C.L., Wang, C.H., Gao, Z.M., Huang, S.M.: Applications of lexical information for algorithmically composing multiple-choice cloze items. In: Proceedings of the Second Workshop on Building Educational Applications Using Natural Language Processing, Ann Arbor, Michigan, U.S., Association for Computational Linguistics (June 2005) 1–8
5. Brown, J., Frishkoff, G., Eskenazi, M.: Automatic question generation for vocabulary assessment. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Vancouver, British Columbia, Canada, Association for Computational Linguistics (October 2005) 819–826
6. Chen, C.Y., Liou, H.C., Chang, J.S.: Fast: An automatic generation system for grammar tests. In: Proceedings of the COLING/ACL on Interactive presentation sessions, Morristown, NJ, U.S., Association for Computational Linguistics (2006) 1–4
7. Lee, J., Seneff, S.: Automatic generation of cloze items for prepositions. In: Proceedings INTERSPEECH 2007, Antwerp, Belgium (August 2007) 2173–2176
8. Kunichika, H., Urushima, M., Hirashima, T., Takeuchi, A.: A computational method of complexity of questions on contents of english sentences and its evaluation. In: ICCE 2002: Proceedings of the International Conference on Computers in Education. (2002) 97–101
9. Segler, T.M.: Investigating the Selection of Example Sentences for Unknown Target Words in ICALL Reading Texts for L2 German. PhD in Informatics, School of Informatics, University of Edinburgh, Edinburgh, U.K. (2005)
10. Terada, H., Tanaka-Ishii, K.: Sorting texts by relative readability. In: Proceedings of Empirical Methods on Natural Language Processing (EMNLP) 2008, Honolulu, Hawaii, U.S., Association for Computational Linguistics (October 2008) 127–133
11. Miyazaki, Y., Norizuki, K.: Developing a computerized readability estimation program with a web-searching function to match text difficulty with individual learners' reading ability. In: Proceedings of WorldCALL 2008, Fukuoka, Japan, CALICO (August 2008) d–111

12. H.Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. The Morgan Kaufmann Series in Data Management Systems, Jim Gray, Series Editor (October 1999)
13. Coniam, D.: A preliminary inquiry into using corpus word frequency data in the automatic generation of english language cloze tests. CALICO Journal 16(2-4) (1997) 15–33
14. Hoshino, A., Huan, L., Nakagawa, H.: A framework for automatic generation of grammar and vocabulary questions. In: Proceedings of WorldCALL 2008, Fukuoka, Japan, WorldCALL (August 2008)

Appendix. Train data and test data

Table 6: Sample questions with different CRR (Correct Response Rates) in (%)

CRR	instance (TOEIC preparation questions)
98.5	If you would like to learn more about [ ] to use this advanced copy machine, simply call the number on the front of the pamphlet and we will send out one of our representatives. a. how b. which c. who d. what
64.1	Workers must [ ] the parcels on to a conveyor belt that carries them to the delivery trucks. a. load b. wrap c. fill d. enter
50.0	Contract negotiations between the union and Pacific Shipping Inc. [ ] in Long Beach after a three-week break. a. have resumed b. has resumed c. is resumed d. resumes
9.7	The purchasing manager is trying to [ ] a deal with the supplier, which could reduce the total cost of materials significantly. a. strike b. discount c. place d. drive

Table 7: Sample questions with predicted difficulty value and CRR in the subject experiment

difficulty	CRR	instance (automatically-generated questions)
-2.95	33.3	He was discharged after the hospital cited no external problems and [ ]. a. negative CT scan results b. results scan negative CT c. results scan negative concussion d. negative concussion scan results
-2.33	0.4	The researchers found for the first time that [ ] pylori reduces the risk of a relapse of stomach cancer by two-thirds. a. removing helicobacter b. removing gastritis c. to remove helicobacter d. to remove gastritis
-1.36	1.0	Koumura later told reporters that Rice's response to his question on North Korea was what Japan [ ] expected. a. had b. has c. was d. were
-1.28	1.0	However, disagreement persisted over which side should act first [ ] the fighting. a. to stop b. stop c. to pull d. pull

MathNat - Mathematical Text in a Controlled Natural Language

Muhammad Humayoun and Christophe Raffalli

Laboratory of Mathematics (LAMA)
Université de Savoie, France
{mhuma, raffalli}@univ-savoie.fr*

Abstract. The MathNat[1] project aims at being a first step towards automatic formalisation and verification of textbook mathematical text. First, we develop a controlled language for mathematics (CLM) which is a precisely defined subset of English with restricted grammar and dictionary. To make CLM natural and expressive, we support some complex linguistic features such as anaphoric pronouns and references, rephrasing of a sentence in multiple ways producing canonical forms and the proper handling of distributive and collective readings.
Second, we automatically translate CLM into a system independent formal language (MathAbs), with a hope to make MathNat accessible to any proof checking system. Currently, we translate MathAbs into equivalent first order formulas for verification.
In this paper, we give an overview of MathNat, describe the linguistic features of CLM, demonstrate its expressive power and validate our work with a few examples.

Key words: Mathematical Discourse, Informal Proofs, Anaphora, Controlled Language, Formalisation

1 Introduction

Since Euclid, mathematics is written in a specific scientific language which uses a fragment of a natural language (NL) along with symbolic expressions and notations. This language is structured and semantically well-understood by mathematicians but still not precise enough for automatic formalisation. By "not precise enough", we mean:

- Like any natural language text, mathematical text contains complex linguistic features such as anaphoric pronouns and references, rephrasing of a sentence in multiple ways producing canonical forms, proper handling of distributive and collective readings, etc.

* This work is funded by "Informatique, Signal, Logiciel Embarqué" (ISLE), Rhone-Alpes, France. http://ksup-gu.grenet.fr/isle/

[1] http://www.lama.univ-savoie.fr/~humayoun/phd/mathnat.html

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 293-307

Received 27/11/09
Accepted 16/01/10
Final version 10/03/10

– To make text comprehensive and aesthetically elegant, mathematicians tend to omit obvious details. Such reasoning gaps may be quite easy for a human to figure out but definitely not trivial for a machine.

In the current state of art, mathematical texts are sometimes formalised in very precise and accurate systems using specific formalisms normally based on some particular calculus or logic. Such a formal piece of mathematics does not contain natural language elements at all. Instead, it contains a lot of technical details of the underlying formal system, making it not suitable for human comprehension. This wide gap between textbook and formal mathematics, reduces the usefulness of computer assisted theorem proving in learning, teaching and formalising mathematics.

In this paper we focus on the first difficulty by developing a controlled language with the look and feel of textbook mathematics. We name it CLM (Controlled Language of Mathematics). It is a precisely defined subset of English with a slightly restricted grammar and lexicon. To make it natural and expressive enough, we support the above mentioned linguistic features. Here are the three main components of MathNat:

1. **The Controlled language of Mathematics (CLM):** Translate the NL text into an abstract syntax tree (AST) in two steps:
   (a) **Sentence Level Grammar:** It is sentence level (without context), attributed grammar with dependent records which we implement in Grammatical Framework (GF) [11]. We describe it in section 3 and 4.
   (b) **Context building:** We build context from the CLM discourse and solve the above mentioned linguistic features. Context building is described in section 5 and the linguistic features are described in section 6.
2. **Translation to MathAbs:** We automatically translate the AST into a system independent formal language (MathAbs), with a hope to make MathNat accessible to any proof checking system, described in section 7.
3. **Proof checking:** Currently, we translate MathAbs into equivalent first order formulas for automatic verification by automated theorem provers (ATP), described in section 7. This step is problematic since most ATP cannot verify even very simple proofs without help from the user. Further, (1) ATP are very sensitive to such hypotheses or details whose sole purpose is to offer an explanation to the reader. (2) Proofs in NL never give the exact list of hypotheses and definitions necessary at each step. This paper does not cover these problems.

The overall picture of the MathNat project is shown in figure 1.

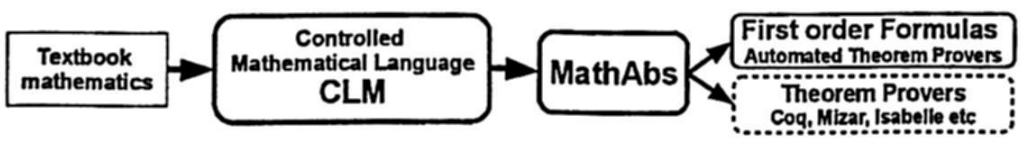

Fig. 1. MathNat - Overall picture

Such a controlled language is definitely easier to read than a formal language used by proof assistants. But is it easier to write? A realistic answer is negative

because a writer may go out of the scope of our grammar very quickly. However, an ambitious answer could be positive. Because the design of CLM supports incremental extendability of the grammar. Further, appropriate tools such as word completion, etc could help the writer to remain in the scope of the grammar.

But even if it fails to give enough freedom to an author for writing mathematics using CLM, we can still consider this work as a first step towards an almost complete mathematical language parser. Although, further work will be needed to extend the coverage tremendously, resolve more linguistic features and solve complexity issues that will certainly arise. With this in mind, the reader should therefore consider this article as a "proof of concept".

2 The Language of textbook mathematics

Mathematical texts mainly consists of theorems, lemmas and their proofs along with some supporting axioms, definitions and remarks. Axioms and definitions normally consist of a few statements expressing a proposition or a definitional equality. On the other hand, a proof is a collection of arguments presented to establish the truth of a proposition. It mainly follows a narrative style and its structure mostly remains the same for all mathematical domains.

The text in figure 2 is the main example that we'll use in this paper to illustrate the possibilities of MathNat. Sentences are numbered for future references. Here are a few remarks, general for mathematical text, showing that this text is already quite hard to formalise automatically: it mixes NL with symbolic expressions; it uses anaphoric pronouns. e.g. at line 7, 12; the use of explicit, implicit references or both e.g. at line 9, 10 and 6 respectively; a lot of keywords are used in the text (e.g. *let, suppose that, then, thus,* etc) that are mostly part of specific patterns such as:"if *proposition* then *proposition*", "(let | suppose | Thus | ...) *proposition* (because | by | ...) *proposition*"; the use of subordinates. e.g. at line 8, noun adjuncts. e.g. "*with no common factor*" at line 4 and explicit quantification. e.g. "for every x, if x is even then $x+1$ is odd" or "there is an integer x such that $x > 0$"; ...

1. **Definition 1.** x is a rational number if it is expressed as $\frac{p}{q}$, where p and q are integers with $q > 0$. [...]
2. **Theorem 1.** *Prove that* $\sqrt{2}$ *is irrational.*
3. **Proof.** Assume that $\sqrt{2}$ is a rational number.
4. By the definition of rational numbers, we can assume that $\sqrt{2} = \frac{a}{b}$ where a and b are non-zero integers with no common factor.
5. Thus, $b\sqrt{2} = a$.
6. Squaring both sides yields $2b^2 = a^2$ (1).
7. a^2 is even because it is a multiple of 2.
8. So we can write $a = 2c$, where c is an integer.
9. We get $2b^2 = (2c)^2 = 4c^2$ by substituting the value of a into equation 1.
10. Dividing both sides by 2, yields $b^2 = 2c^2$.
11. Thus b is even because 2 is a factor of b^2.
12. If a and b are even then they have a common factor.
13. It is a contradiction.
14. Therefore, we conclude that $\sqrt{2}$ is an irrational number.
15. This concludes the proof. □

Fig. 2. A typical math text

3 Sentence level CLM Grammar

GF [11] is a programming language for defining NL grammars that is based on Martin-Löf's dependant type theory [9]. We refer to [5] for further details. In GF,

we completely ignore the context, and design the CLM as an attributed grammar with dependant records. A GF grammar has two components: *abstract syntax* and *concrete syntax*. *Abstract syntax* defines semantic conditions to form abstract syntax trees (AST) of a language with grammatical functions (*fun*) making nodes of categories (*cat*). While a *concrete syntax* is a set of linguistic objects (strings, inflection tables, records) associated to ASTs, providing rendering and parsing. The process of translating an AST into one of its linguistic objects is called *linearization*.

```
cat Subject; Attribute; Prop;
fun MkProp: Subject -> Attribute -> Prop;
fun MkPronSubj: Pron -> Subject;
fun It: Pron; They: Pron;
```

Fig. 3. abstract syntax

Consider a part of our grammar for propositions such as "they are even integers". In figure 3, we define three categories. The function `MkProp` takes two parameters (a subject and an attribute) and forms a proposition. A subject is formed by pronouns `It` or `They`.

```
cat Property ; Type ;
fun MkAttrb: [Property] -> Type -> Attribute;
fun Even     : Property ;
fun Integer : Type ;
```

Fig. 4. abstract syntax

As shown in figure 4, `MkAttrb` function forms an attribute with a list of `Property` and `Type`. Next, we define functions `Even` and `Integer` of category `Property` and `Type` respectively. In full CLM grammar, we add properties (e.g. positive, odd, distinct, equal, etc) and types (e.g. rational, natural number, set, proposition, etc) in a similar fashion. To map this abstract syntax into its concrete syntax, we define a set of linguistic objects corresponding to the above categories and functions.

```
lincat Property   = {s : Str};
lin    Even       = {s ="even"};
param  Number = Sg | Pl ;
lincat Type   = {s : Number => Str};
lin Integer = {s = table{
                     Sg => "integer";
                     Pl => "integers"}};
lincat Pron = {s : Str ; n : Number} ;
lin    It   = {s = "it"   ; n = Sg};
lin    They = {s = "they" ; n = Pl} ;
```

Fig. 5. concrete syntax

In figure 5, the first line defines the linearization of `Property` category which is simply a string record. The second line shows this fact for the linearization of its function `Even`. The linearization of category `Type` is an inflection table (a finite function) from number to string, having one string value for each (singular and plural) as shown in fourth line. Its function `Integer` fills this table with two appropriate values in the next three lines.

```
lincat Attrb = {s : Number => Str};
lin MkAttrb props type = {s = table {
      Sg => artIndef ++ props.s ++ type.s!Sg;
      Pl =>             props.s ++ type.s!Pl}};
```

Fig. 6. concrete syntax

The linearization of pronoun `Pron` is a record, having a string and number. Further, we define the linearization of its functions and mention the fact that `It` is singular and `They` is plural, which will help us to make number agreement. Similar to category `Type`, `Attribute` is also an inflection table. Therefore, in figure 6, we define the linearization of its function `MkAttrb` accordingly. For instance, for singular value, we select the string value

of the category list of `Property` (props) with `(.s)`. Then, we select the singular string value of category `Type` with `(type.s!Sg)`. `(++)` concatenates these two strings with a space between them. `artIndef` makes an agreement for an indefinite article with the first letter of next word. e.g. producing *"an even number"* or *"a positive number"*. It is defined in GF resource library [12] which provides basic grammar for fourteen languages as an API.

Similarly, in figure 7, to form a proposition, in `MkProp`, we select appropriate string values of tables `be` and attribute by an agreement of number with subject, and concatenate them with subject. For instance, if we parse the propositions such as "they are even integers" and "it is an even integer", we'll get following **abstract syntax trees:**

```
lincat Prop = {s : Str};
oper be = {s = table{Sg => "is" ; Pl => "are"}};
lin MkProp subj attrb =
  {s = subj.s ++ be.s!subj.n ++ attrb.s!subj.n};
```

Fig. 7. concrete syntax

```
1. MkProp (MkPronSubj They) (MkAttrb (BaseProperty Even) Integer)
2. MkProp (MkPronSubj It) (MkAttrb (BaseProperty Even) Integer)
```

Fig. 8. abstract syntax trees

4 Synopsis of CLM Grammar

As a whole, CLM text is a collection of axioms, definitions, propositions (theorems, lemma, ...) and proofs structured by specific keywords (Axiom, Definition, Theorem and Proof). The text following these keywords are list of sentences obeying different GF grammars (one for axioms, one for definition, ...). These grammars are not independent and share a lot of common rules.

We present in this section a short but incomplete synopsis of the grammar for sentences allowed in theorems and proofs. We describe it in an abstract way with some examples and it obeys the following conventions: [text] means that *text* is optional, (text1|text2) means that both *text1* and *text2* are possible, dots (...) means that only few constructions are given due to space limitation, and each dash (–) represents a pattern in the grammar. We start this synopsis by extending the grammar of our running example:

1. Exp is a symbolic expression (equation not allowed).
   It is encapsulated by quotes to distinguish it from natural language parts of grammar. Defining a formal grammar for symbolic expressions and equations in GF is definitely possible but as it is specially designed for NL grammars, in past, it has caused serious efficiency overhead for parsing, because of CLM's size. Therefore we define a *Labelled BNF grammar* in bnfc tool [4]. Examples of Exp are $\sqrt{2}$ in line 2-3, x in line 1 and a^2 and 2 in line 7 of figure 2.
2. Exps is a list of Exp. e.g. a, b in line 12 of figure 2, etc.
3. `Subject`, as partially described before, is (Exps | anaphoric pronouns | ...)
4. `Attribute`, as partially described before, is (

`Quantity`, list of `Property` and `Type`. e.g. two positive even integers. three irrational numbers, etc | `Property` e.g. positive, even, distinct, equal, etc | `Quantity`, `Relation` and `Exp`. e.g. an element of y. two elements of z, etc |...)

5. `Prop` of *proposition* is (Positive and negative statements formed by `Subject` and `Attribute` e.g. at line 2, 3, etc | Statements formed by `Subject`, `Quantity` and `Relation`. e.g. x, y and z have a common factor, they have two common multiples, etc | Statements containing existential and universal quantification. | If then statements. | Disjunction statements. | ...)
6. `Eq` is a symbolic equation, encapsulated by quotes. As described for `Exp`, it is also defined as a *Labelled BNF grammar*. e.g. in line 4, 5, 6, 8, 9, 10, etc
7. `EqOrProp` is (`Eq` | `Prop`)
8. `DefRef` is (`Property` and `Type`. e.g the definition of even numbers, etc | `Property` e.g. the definition of evenness, etc)
9. `Operation` is (`Relation` e.g factoring both sides, taking square at both sides, etc | `Relation` and `Exp` e.g. dividing both sides by x, multiplying the equation by 2, etc | ...)
10. `Justif` is (`Eq` | `Prop` | `DefRef` | `Operation`)

Sentence for theorems could be described as follows:

11. *"Prove statement" with an optional subordinate*
- [(show|prove) [that]] `Eq` [holds] [(where|when|if|with a condition that|...) `EqOrProp`]
- [(show|prove) that] `Prop` [(where|when|if|with a condition that|...) `EqOrProp`]

e.g. "Prove that $x + y = x$ holds if $y = 0$", "If x is even then $x + 1$ is odd with a condition that $x > 0$", line 2 of figure 2, etc.
The main difference between these two patterns is "holds" which is optional for statements containing an equation, but does not appear in statements containing propositions. This applies to all CLM statements. So, in the following statements we mention these two patterns as one using `EqOrProp`

12. *Assumption with an optional subordinate*
- (let |[we] suppose [that]|we can write [that]|...) `EqOrProp` [holds] [(where|...) `EqOrProp`] e.g. "we suppose that $x + y > 0$", line 3 of figure 2, etc

Note: with the example of section 3, we can infer a statement such as "let x is an even number", which is grammatically incorrect. In fact, the actual grammar defined for propositions is a bit more complicated than what is given in section 3. In the actual grammar, attribute and proposition are inflection tables with two values; one for *let* statements ("be an even..."), and second for the remaining ("is/are ...").

13. *Sentences that cannot be the first sentence in a theorem and proof*
(then|thus|so|therefore), followed by any statement. e.g. line 5, 11 of figure 2, etc

Note: In theorem, statements of the form 12 and 13 are often stated to help the reader as a starting point for the proof.

Proof statements could be described as follows:

14. Shall-Prove statement with an optional subordinate
- we [(will | shall |have to)] prove [that] `EqOrProp` [holds] [(where|...) `EqOrProp`]

Shall-Prove pattern is almost the same as 11, but in proofs, with different keywords, we distinguish *goals* from *deductions.*

15. Assumption with an optional subordinate
Same as 12. e.g. line 8 of figure 2 (if we remove "So" from the sentence), etc

16. Assumption with a justification and an optional subordinate
- ([we] assume [that]|...) `EqOrProp` (since |because |by |...) `Justif` [(where |...) `EqOrProp`]
- (since |because |by |...) `Justif` ([we] assume [that]|...) `EqOrProp` [(where |...) `EqOrProp`] e.g. line 4 of figure 2, etc

These patterns are rough estimate of the actual coverage, it is possible to infer a statement which is grammatically incorrect. e.g. "assume $x + y = z$ because squaring both sides". In actual grammar, we have six patterns for 16 that ensures the grammatical correctness. In doing so, `Justif` is not just one category. Instead it is formed by some of these: (`Eq` |`Prop` |`DefRef` |`Operation`). This applies to all patterns of the grammar.

17. Deduction with an optional subordinate
- [(we (get |conclude |deduce |write |...) [that])] `EqOrProp` [holds] [(where|...) `EqOrProp`]

e.g. "we conclude that there is an even number x such that $x > 2$", line 5, 12 and 14 (if we remove "Therefore,") of figure 2, etc

18. Deduction with a justification and an optional subordinate
- (we get |...) `EqOrProp` (because |...) `Justif` [(where |...) `EqOrProp`]
- (because |...) `Justif` (we get |...) `EqOrProp` [(where |...) `EqOrProp`]
- `Operation` (yields | shows | establishes |...) `EqOrProp` [(where |...) `EqOrProp`]
- (because |...) `Justif`, `EqOrProp` [where `EqOrProp`]

e.g. line 6, 7, 9, 10, 11 (if we remove "Thus") of figure 2, etc

19. Proof by case (nested cases are allowed)

we proceed by case analysis	if *condition* then ...
case: *condition* ... [this ends the case.]	otherwise if *condition* then ...
case: *condition* ... [this ends the case.]	.....
..... [it was the last case.]	otherwise ...

5 Discourse Building

GF can compile the grammar into code usable in other general purpose programming languages. For instance, in Haskell, rendering functions *linearize* and

parse are made available as ordinary functions. It translates abstract syntax into algebraic data types forming objects of type AST.

When the math text is parsed by CLM parser, a list of sentence level AST is produced. We recognise each AST by pattern matching on algebraic data types and build the context from CLM discourse. For an AST, we record every occurence of symbolic expressions, equations, pronouns and references as shown in Table 1. Further from Table 1, it seems like we keep a placeholder for each anaphoric pronoun that apears in the text. In fact, they are resolved immediately as they apear with the algorithm described in section 6. Context building and translation of math text into MathAbs are interleaved, as shown in the following procedure. So both are performed in the same step. However MathAbs translation is described in section 7. Context for theorem and proof in figure 2 is given in Table 1, 2 and 3.

Notations: S denotes an arbitrary AST (abstract syntax tree).
Context is 3-tuple `(CV, ST, CE)` where:
`CV` is a list of 4-tuple *(sentence number S_n, list of symbolic objects ($obj_1, ..., obj_n$), type, number)* as shown in table 1.
`ST` is a list of 3-tuple *(S_n, logical formula, type)* as shown in table 2.
`CE` is a list of 3-tuple *(S_n, equation, reference)* as shown in table 3.

Procedure: we start the analysis with an empty *Context* which evolves while examining the AST of each sentence. The following procedure is repeated until we reach the end of the text. Just a few cases are mentioned here due to the space limitations.

```
//an assumption or deduction containing an equation. e.g. line 4,8 or 5,6,9 respectively
If S matches (Assume Eq) or (Deduce Eq)
 mathabs_eq := translate Eq into MathAbs's formula
 left_eq := left side of mathabs_eq*
 lookup for a latest (S_n, obj, type, Sg) of CV if left_eq=obj
   t := if such (S_n, obj, type, Sg) found then type else NoType
 add (S_n, left_eq, t, Sg) in CV
 add (S_n, mathabs_eq, reference if provided in Eq) in CE
 add (S_n, mathabs_eq, st_type(S)) in ST
```

```
st_type(S) := if S matches (Assume ...) then Hypothesis //common to all
              else if S matches (Deduce ...) then Deduction else Goal
```

*The choice of taking the left side of an equation is a bit naive. But there is no established algorithm to decide which identifier an anaphoric pronoun refers to. e.g. In "[...] we have $\forall_x(f(x) = p)$. Thus, it is a constant function" pronoun "it" refers to f, which is not at all trivial even for a human sometimes. So we always take the left side as a convention, which in fact makes sense for many equations.

```
//an assumption or deduction containing Exps e.g. line 2,3 or line 7,11 respectively
If S matches (Assume (MkProp ... Exps)) or (Deduce (MkProp ... Exps))
1. Statements** such as "we (assume|conclude) that x is a positive integer"
 type := given in S
 number := if (length Exps)=1 then Sg else Pl
```

```
add (S_n, Exps***, type, number) in CV
mathabs_exp := translate S into MathAbs's formula
add (S_n, mathabs_exp, st_type(S)) in ST
```

**We mention only one case due to space limitation.

***The convention of taking whole expression is also a bit naive. But again, there is no established algorithm to decide which identifier an anaphoric pronoun refers to. e.g. "[...] because $a|p$ (a divides p), it is a prime or 1" Here pronoun "it" refers to a. But in statement "[...] $2|b$. Thus it is even" pronoun it refers to b.

```
//a prove statement containing an equation. e.g. show that 2x + 2y = 2(x + y)
If S matches (Prove Eq)
 mathabs_eq := translate Eq into MathAbs's formula
 left_eq := left side of mathabs_eq
 add (S_n, left_eq , NoType, Sg) in CV
 add (S_n, mathabs_eq, reference if provided in Eq) in CE
 vars := list of all variables appeared in mathabs_formula
 not_decl_vars := all variables of vars that not found in MathAbs context
 //e.g. 2x + 2y = 2(x + y) translated as ∀x,y(2x + 2y = 2(x + y)) if x, y not found
 mathabs_formula := ∀not_decl_vars(mathabs_eq)
 add (S_n, mathabs_formula, st_type) in ST
```

6 Linguistic features

Recall that (CV, ST, CE) is the *Context* defined in section 5. Also recall that in the course of context building, we resolve all kind of anaphora immediately as they apear with the following algorithm.

6.1 A Naive Algorithm for Anaphoric Pronouns

If S matches (...(...It) ...)
i.e. a statement containing pronoun "it". We replace this pronoun with the latest obj_1 of *(S_n, obj_1, Sg, type)* from CV. e.g. line 7 of figure 2 is interpreted as "a^2 is even because a^2 is a multiple of 2"

If S matches (...(...They) ...)

1. If no *quantity* (e.g. two, three, ...) is mentioned in S, we replace pronoun "they" with the latest $(obj_1, ..., obj_n)$ of *(S_n, (obj_1, ..., obj_n), Pl, type)* from CV. e.g. line 12 of figure 2 is interpreted as "If a and b are even then a and b have a common factor"
2. Otherwise, if there is a *quantity* Q mentioned in S then we replace this pronoun with the latest $(obj_1, ..., obj_n)$ of *(S_n, (obj_1, ..., obj_n), Pl, type)* when Q=*length*$(obj_1, ..., obj_n)$.
   e.g. the last statement from "suppose that $x+y+z > 0$.[...] assume that a and b are positive numbers.[...] they are three even integers" is interpreted as "[...] x, y and z are three even integers"

6.2 Anaphora of Demonstrative Pronouns

The <*type,number*> pair of *(S_n, (obj_1, ..., obj_n), number, type)* from CV, allows to solve anaphora for demonstrative pronouns "this" and "these". e.g. "*these* integers ...", is replace by the latest (obj_1, ..., obj_n) with *number=Pl* $\wedge$ *type=Integer.*

For the moment we only deal with the pronouns referring to expressions. Pronouns referring to propositions and equations are left as a future work.

Table 1. CV: Symbolic objects

S_n	Objects	Type	Number	S_n	Objects	Type	Number	S_n	Objects	Type	Number
2.	$\sqrt{2}$	NoType	Sg	7.	It	?	Sg	10.	b^2	NoType	Sg
3.	$\sqrt{2}$	Rational	Sg	7.	2	NoType	Sg	11.	b	Integer	Sg
4.	$\sqrt{2}$	Rational	Sg	8.	a	Integer	Sg	11.	2	NoType	Sg
4.	a, b	Integer	Pl,2	8.	c	Integer	Sg	11.	b^2	NoType	Sg
5.	$b\sqrt{2}$	NoType	Sg	9.	$2b^2$	NoType	Sg	12.	a, b	Integer	Pl,2
6.	$2b^2$	NoType	Sg	9.	a	Integer	Sg	12.	They	?	Pl,?
7.	a^2	NoType	Sg	10.	2	NoType	Sg	14.	$\sqrt{2}$	Irrational	Sg

Table 2. ST: Logical formulas

S_n	Logical formula	Stmnt type	S_n	Logical formula	Stmnt type
2.	$\sqrt{2} \notin Q$	Goal	7.	even(a^2)	Deduction
3.	$\sqrt{2} \in Q$	Hypothesis	8.	$c \in Z \wedge a = 2c$	Hypothesis
4.	$a, b \in Z \wedge$		9.	$2b^2 = (2c)^2 = 4c^2$	Deduction
	positive(a) $\wedge$		10.	$b^2 = 2c^2$	Deduction
	positive(b)$\wedge$	Hypothesis	11.	even(b)	Deduction
	no_cmn_factor(a, b)		12.	even(a)$\wedge$ even(b) $\Rightarrow$	Deduction
	$\wedge\sqrt{2} = a/b$			one_cmn_factor(a, b)	
5.	$b\sqrt{2} = a$	Deduction	13.	False	Deduction
6.	$2b^2 = a^2$	Deduction	14.	$\sqrt{2} \notin Q$	Deduction

Table 3. CE: Equations

S_n	Equation	Reference	S_n	Equation	Reference	S_n	Equation	Reference
4.	$\sqrt{2} = a/b$	NoRef	6.	$2b^2 = a^2$	1	9.	$2b^2 = (2c)^2 = 4c^2$	NoRef
5.	$b\sqrt{2} = a$	NoRef	8.	$a = 2c$	NoRef	10.	$b^2 = 2c^2$	NoRef

6.3 Solving References

1. Explicit reference to an equation as appeared at line 9. In the current settings of the context, it is trivial to solve such anaphora because each reference is preserved. e.g. line 9 is interpreted as "We get $2b^2 = (2c)^2 = 4c^2$ by substituting the value of a into $2b^2 = a^2$"
2. Implicit reference to an equation as appeared at line 6 and 10. At line 10, dividing both sides by 2 implies that there is an equation in some previous sentence. So we check this condition and if an equation is found in CE, we put it in place and interpret it as "Dividing $2b^2 = (2c)^2 = 4c^2$ by 2 at both sides, yields $b^2 = 2c^2$"

3. Reference to an equation that is mentioned far from the current sentence. e.g. "dividing the (last|first) equation by 2". It is also quite trivial to solve because we lookup CE for last or first element.
4. Reference to a hypothesis, deduction or statement. e.g. "we deduce that $x + y = 2(a + b)$ by the (last|first) (statement|hypothesis|deduction)".
   For instance, for a statement containing "by the last hypothesis", we simply pick the latest *formula* of *(S_n, formula, type)* from ST when *type=Hypothesis.* However, for a statement containing "by the last statement", we pick the latest *formula* of *(S_n, formula, type)* when *type=(Hypothesis or Deduction).*

6.4 Distributive vs. Collective Readings

Like any natural language text, distributive and collective reading are common in math text and we deal with them in CLM appropriately. For example, the statements such as "x, y are positive" and "x, y are equal" are distributive and collective respectively. Some collective readings could be ambiguous. Consider the statement "a,b,c are distinct". Are variables a, b, c pair-wise distinct or just some of them distinct? Currently we neglect such ambiguity and always translate such properties as 2-arity predicates. Further, because collective readings require their subjects to be plural, statements such as "x is equal" are not allowed.

7 MathAbs and proof checking

The Haskell code supporting above mentioned linguistic features does more. It translates the AST given by GF to an abstract Mathematical Language (MathAbs). This language is a system independent formal language that we hope will make MathNat accessible to any proof checking system.

MathAbs (formally new_command) was designed as an intermediate language between natural language proofs and the proof assistant PhoX [10] in the DemoNat project [14][15]. Since new_command includes some of the standard commands of PhoX, we adapt it to MathNat by doing some minor changes.

MathAbs can represent theorems and their proofs along with supporting axioms and definitions. On a macro level, a MathAbs's document is a sequence of definitions, axioms and theorems with their proofs. However, analogous to informal mathematical text, the most significant part of MathAbs is the language of proof. While, the definitions and statements of propositions are a fraction of the text compared to the proofs.

A proof is described as a tree of logical (meta) rules. Intuitively, at each step of a proof there is an implicit active sequent, with some hypotheses and one conclusion, which is being proved and some other sequents to prove later. The text in NL explains how the active sequent is modified to progress in the proof and gives some justifications (hints). Similarly, a thereom forms the initial sequent with some hypotheses and one goal, which is then passed to the proof. While, axioms and definitions also form the intial sequent by adding them as hypotheses.

The basic commands of MathAbs are let to introduce a new variable in the sequent, assume to introduce a new hypothesis, show to change the conclusion of the sequent, trivial to end the proof of the active sequent and { ...; ... } to branch in the proof and state that the active sequent should be replaced by two or more sequents. deduce A ... is a syntactic sugar for { show A trivial ; assume A ... }. We translate math text of figure 2 in MathAbs as shown below:

1. `Definition.` $r \in Q \Leftrightarrow \exists_{p,q \in Z}(r = \frac{p}{q} \wedge q > 0)$
2. `Theorem.` show $\sqrt{2} \notin Q$
3. `Proof.` assume $\sqrt{2} \in Q$
4. let $a, b \in$ Z assume $\sqrt{2} = a/b$ assume `positive(`a`)` $\wedge$ `positive(`b`)` $\wedge$ `no_cmn_factor(`a, b`)` `by def rational_Number`
5. deduce $b\sqrt{2} = a$
6. deduce $2b^2 = a^2$ `1 by oper squaring_both_sides(`$b\sqrt{2} = a$`)`
7. deduce `multiple_of(`a^2`, 2)`
   deduce `even(`a^2`)` `by form multiple_of(`a^2`, 2)`
8. let $c \in$ Z assume $a = 2c$
9. deduce $2b^2 = (2c)^2 = 4c^2$ `by oper substitution(`$a, 2b^2 = a^2$`)`
10. deduce $b^2 = 2c^2$ `by oper division(`$2, 2b^2 = (2c)^2 = 4c^2$`)`
11. deduce `factor_of (2,` b^2`)`
    deduce `even(`b`)` `by form factor_of(2,`b^2`)`
12. deduce `(even(`a`)` $\wedge$ `even(`b`))` $\Rightarrow$ `one_cmn_factor(`a`,` b`)`
13. show $\perp$ trivial

Remarks:

- At line 7, first, we deduce the justification i.e. `multiple_of(`a^2`, 2)`, and then deduce the whole statement. Same applies to 11.
- However, the above rule does not apply to definitional references and operations as shown in 4, 6, 9, 10.
- We can safely ignore Line 14 and 15 of figure 2 because the proof tree is already finished at line 13.

We can represent above MathAbs proof as a proof tree using arbitrary rules (not just the rules of natural deduction). Then, for each rule we can produce a formula that justifies it. The line 2-5 of above MathAbs can be read as the following proof tree:

$$\dfrac{\dfrac{\dfrac{}{\Gamma_2 \vdash b\sqrt{2} = a} \quad \dfrac{\vdots}{\Gamma_3 \equiv (\Gamma_2, b\sqrt{2} = a) \vdash \sqrt{2} \notin Q}}{\Gamma_2 \equiv (\Gamma_1, a, b \in \mathtt{Z},\ \sqrt{2} = a/b,\ \mathtt{positive}(a),\ \mathtt{positive}(b),\ \mathtt{no\_cmn\_factor}(a, b)) \vdash \sqrt{2} \notin Q}}{(\Gamma_1 \equiv \Gamma, \sqrt{2} \in Q) \vdash \sqrt{2} \notin Q}}{\Gamma \vdash \sqrt{2} \notin Q}$$

Fig. 9. MathAbs proof as a proof tree. Γ is a context which contains the usefull definitions and axioms, needed to validate this proof.

As a first prototype we implemented a translation from MathAbs to first order formulas for validation. The above MathAbs is translated as follows (one formula for each rule):

3. $\vdash$ (rational($\sqrt{2}$) $\Rightarrow$ irrational($\sqrt{2}$)) $\Rightarrow$ irrational($\sqrt{2}$)
4. $\Gamma_1 \vdash \forall_{a,b}$((int($a$) $\wedge$ int(b) $\wedge \sqrt{2} = a/b \wedge a, b > 0 \wedge$ gcd(a, b) $= 1$) $\Rightarrow$ irrational($\sqrt{2}$)) $\Rightarrow$ irrational($\sqrt{2}$)
   where $\Gamma_1 \equiv$ rational($\sqrt{2}$)
5. $\Gamma_2 \vdash (b\sqrt{2} = a \wedge (b\sqrt{2} = a \Rightarrow$ irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
   $\Gamma_2 \vdash b\sqrt{2} = a$
   where $\Gamma_2 \equiv \Gamma_1, \forall a, b$ (int(a)$\wedge$int(b) $\wedge \sqrt{2} = a/b \wedge a, b > 0 \wedge$ gcd(a, b) $= 1$)
6. $\Gamma_3 \vdash (2b^2 = a^2 \wedge (2b^2 = a^2 \Rightarrow$ irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
   $\Gamma_3 \vdash 2b^2 = a^2$
   where $\Gamma_3 \equiv \Gamma_2, b\sqrt{2} = a$
7. $\Gamma_4 \vdash$ (multiple_of($a^2, 2$) $\wedge$ (multiple_of($a^2, 2$) $\Rightarrow$ irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
   $\Gamma_4 \vdash$multiple_of($a^2, 2$)
   where $\Gamma_4 \equiv \Gamma_3, 2b^2 = a^2$
   $\Gamma_5 \vdash$ (even(a^2) $\wedge$ (even(a^2) $\Rightarrow$irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
   $\Gamma_5 \vdash$ even(a^2)
   where $\Gamma_5 \equiv \Gamma_4$, multiple_of($a^2, 2$)
8. $\Gamma_6 \vdash \forall c$((int($c$) $\wedge\ a = 2c$) $\Rightarrow$irrational($\sqrt{2}$)) $\Rightarrow$ irrational($\sqrt{2}$)
   where $\Gamma_6 \equiv \Gamma_5$, even(a^2)
9. $\Gamma_7 \vdash (2b^2 = (2c)^2 = 4c^2 \wedge (2b^2 = (2c)^2 = 4c^2 \Rightarrow$irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
   $\Gamma_7 \vdash 2b^2 = (2c)^2 = 4c^2$
   where $\Gamma_7 \equiv \Gamma_6$, int(c) $\wedge\ a = 2c$
10. $\Gamma_8 \vdash (b^2 = 2c^2 \wedge (b^2 = 2c^2 \Rightarrow$ irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
    $\Gamma_8 \vdash b^2 = 2c^2$
    where $\Gamma_8 \equiv \Gamma_7, (2b^2 = (2c)^2 = 4c^2)$
11. $\Gamma_9 \vdash$ (factor_of($2, b^2$)$\wedge$(factor_of ($2, b^2$) $\Rightarrow$ irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
    $\Gamma_9 \vdash$ factor_of($2, b^2$)
    where $\Gamma_9 \equiv \Gamma_8, (b^2 = 2c^2)$
    $\Gamma_{10} \vdash$ (even(b)) $\wedge$ (even(b) $\Rightarrow$ irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
    $\Gamma_{10} \vdash$ even(b)
    where $\Gamma_{10} \equiv \Gamma_9$, factor_of ($2, b^2$)
12. $\Gamma_{11} \vdash$ ((even(a)$\wedge$ even(b) $\Rightarrow$ one_cmn_factor(a, b)) $\wedge$ ((even(a)$\wedge$ even(b) $\Rightarrow$ one_cmn_factor(a, b)) $\Rightarrow$ irrational($\sqrt{2}$))) $\Rightarrow$ irrational($\sqrt{2}$)*
    $\Gamma_{11} \vdash$ even(a)$\wedge$ even(b) $\Rightarrow$ one_cmn_factor(a, b)
    where $\Gamma_{11} \equiv \Gamma_{10}$, even($b$)
13. $\Gamma_{12} \vdash (\bot \Rightarrow$ irrational($\sqrt{2}$))
    $\Gamma_{12} \vdash \bot$
    where $\Gamma_{12} \equiv \Gamma_{11}$, (even($a$) $\wedge$ even(b) $\Rightarrow$ one_cmn_factor(a, b))

Remarks:

- Since deduce A is a syntactic sugar of { show A trivial ; assume A ... }, it produces a lot of tautologies of the form $(A \wedge (A \Rightarrow B)) \Rightarrow B$ in the first-order formulas. Where B is the main goal to prove. They are marked with * above.
- In proof, if we add in a sentence such as "proof by contradiction" this adds a MathAbs command show $\bot$ that would replace the conclusion irrational($\sqrt{2}$) by $\bot$.
- The justifications such as "by def rational_Number" that are preserved in MathAbs were removed from the first order translation, because most of the automated theorem provers are unable to use such justifications.

– In the above first order formulas, types are treated as predicates. The notion of types used in the grammar is linguistic and do not exactly correspond to the notion of type in a given type theory. This is one of the main problems if we want to translate CLM to a typed framework such as Coq[2] or Agda[3].

8 Related Work

AutoMath [2] of N.d. Bruijn, is one of the pioneering work in which a very restricted proof language was proposed. After that such restricted languages are presented by many systems. For instance, the language of Mizar, Isar [16] for Isabelle, the notion of *formal proof sketches*[17] for Mizar and Mathematical Proof Language *MPL* [1] for Coq. However such languages are quite restricted and non ambiguous having a programming language like syntax, with a few syntactic constructions. Therefore, like MathAbs, we consider them as an intermediate language between mathematical text and proof checking system.

The MathLang project [7] goes one step further by supporting the manual annotation of NL mathematical text. Once the annotation is done by the author, a number of transformations to the annotated text is automatically performed for automatic verification. So MathLang seems quite good in answering the second question raised in the introduction but neglects the possibility of NL parsing completely.

The work of Hallgren et al.[6] presents an extension to the type-theoretical syntax of logical framework Alfa, supporting a self extensible NL input and output. Like MathNat, its NL grammar is developed in GF but it does not support the rich linguistic features as we do. Similar to this work, we hope to make CLM grammar extensible in future.

Nthchecker of Simon [13] is perhaps the poineering work for its time, towards parsing and verifying informal proofs. However, according to Zinn [18] its linguistic and mathematical analysis is quite adhoc and we second his opinion.

In recent times, Vip - the prototype of Zinn [18] is a quite promising work. In his doctoral manuscript and papers, Zinn gives a good linguistic and logical analysis of textbook proofs. In Vip, he builds an extension of discourse representation theory (DRT) for parsing and integrates proof planning techniques for verification. Vip can process two proofs (nine sentences) from number theory. In our opinion, this coverage is too limited to verify the usefulness of presented concepts. Further, it supports limited linguistic features. Unfortunately, no further work has appeared after 2005. We do not use DRT because it is perhaps too complex and an overkill for the task of analysing mathematical text. However, we may consider to integrate with proof planning techniques for verification in future.

Naproche [8] is also based on an extension of DRT. Like MathNat, Naproche translates its output to first order formulas. Currently, it has a quite limited

[2] http://coq.inria.fr/

[3] http://wiki.portal.chalmers.se/agda/

controlled language without rich linguistic features. Further, like MathNat, the problem of reasoning gaps in math text is yet to be tackled.

WebALT[3] is a GF-based project that tries to make multilingual mathematical exercises in seven languages. It has a fairly good coverage for mathematical statements. However, a significant part of mathematics, i.e. the language of proof is out of scope of this work.

9 Conclusion and Future work

For textbook mathematical text, we do not know any system that provides such linguistic features as MathNat does. The coverage of CLM facilitates some common reasoning patterns found in proofs but it is still limited. So for coverage, we aim at working in two directions: enlarging the grammar manauly for common patterns and supporting the ability of being extensible for some parts of CLM grammar.

Context building for symbolic expressions and equations should be improved and we want to find consistent algorithm to pick the right pronoun referents. Some theorems and their proofs from elementary number theory, set theory and analysis were parsed in CLM, and translated in MathAbs, which was further translated in equivalent first order formulas. But we were able to validate only few of them due to the reasoning gaps. So we want to improve the MathNat interaction with automated theorem provers. We also want to explore the possibility of integrating with various proof assistants.

References

1. H. Barendregt. 2003. Towards an Interactive Mathematical Proof Language. Thirty Five Years of Automath, Ed. F. Kamareddine, Kluwer, 25-36.
2. N. G. de Bruijn. 1994. Mathematical Vernacular: a Language for Mathematics with Typed Sets. In R. Nederpelt, editor, selected Papers on Automath, pages 865-935.
3. O. Caprotti. WebALT! Deliver Mathematics Everywhere. In Proceedings of SITE 2006. Orlando March 20-24, 2006.
4. M. Forsberg, A. Ranta. 2004. Tool Demonstration: BNF Converter HW'2004, ACM SIGPLAN.
5. Grammatical Framework Homepage. `http://www.grammaticalframework.org/`
6. T. Hallgren & A. Ranta. 2000. An extensible proof text editor. Springer LNCS/LNAI 1955.
7. F. Kamareddine & J. B. Wells. 2008. Computerizing Mathematical Text with MathLang, ENTCS, 205, 1571-0661, Elsevier Science Publishers B. V. Amsterdam.
8. D. Kühlwein, M. Cramer, P. Koepke, and B. Schröder. 2009. The Naproche System. In Intelligent Computer Mathematics, Springer LNCS, ISBN: 978-3-642-02613-3.
9. Per Martin-Löf. 1984. Intuitionistic Type Theory. Bibliopolis, Napoli.
10. The PhoX Proof Assistant. `http://www.lama.univ-savoie.fr/~RAFFALLI/af2.html`
11. A. Ranta. 2004. Grammatical Framework: A Type-Theoretical Grammar Formalism. Journal of Functional Programming, 14(2):145189.
12. A. Ranta. 2008. Grammars as Software Libraries. To appear in G. Huet, et al. (eds), From semantics to computer science. Cambridge University Press.
13. D.L. Simon. 1988. Checking natural language proofs, in: Springer LNCS 310.
14. P. Thévenon. 2006. PhD Thesis. Vers un assistant à la preuve en langue naturelle. Université de Savoie, France.
15. P. Thévenon. 2004. Validation of proofs using PhoX. ENTCS. `www.elsevier.nl/locate/entcs`
16. M. Wenzel. 1999. Isar: a generic interpretative approach to readable formal proof documents, Springer LNCS 1690.
17. F. Wiedijk. 2003. Formal Proof Sketches. TYPES 2003, Italy, Springer LNCS 3085, 378-393
18. C. Zinn. 2006. Supporting the formal verification of mathematical texts. Journal of Applied Logic, 4(4).

Applications

A Low-Complexity Constructive Learning Automaton Approach to Handwritten Character Recognition

Aleksei Ustimov[1], M. Borahan Tümer[2], and Tunga Güngör[1]

[1] Department of Computer Engineering, Boğaziçi University, İstanbul, Türkiye
[2] Department of Computer Science Engineering, Marmara University, İstanbul, 90210, Türkiye

Abstract. The task of syntactic pattern recognition has aroused the interest of researchers for several decades. The power of the syntactic approach comes from its capability in exploiting the sequential characteristics of the data. In this work, we propose a new method for syntactic recognition of handwritten characters. The main strengths of our method are its low run-time and space complexity. In the lexical analysis phase, the lines of the presented sample are segmented into simple strokes, which are matched to the primitives of the alphabet. The reconstructed sample is passed to the syntactic analysis component in the form of a graph where the edges are the primitives and the vertices are the connection points of the original strokes. In the syntactic analysis phase, the interconnections of the primitives extracted from the graph are used as a source for the construction of a learning automaton. We reached recognition rates of 72% for the best match and 94% for the top five matches.

Key words: Syntactic Pattern Recognition, Handwritten character recognition, 2D Space, Parsing, Automaton, Graph, Data sequence, Expected Value, Matching

1 Introduction

Pattern recognition receives increasing attention since longer than three decades. While considerable progress have been made some challenging problems remain unsolved [1, 5].

Syntactic Pattern Recognition (SPR), one of the four a subcategories of Pattern Recognition consists of two parts: lexical analysis (segmentation) and syntactic analysis (parsing).

In lexical analysis a pattern is reconstructed in terms of primitives[7], that are obtained using deterministic [2] or adaptive [19] way. Syntactic analysis is responsible for grammatical inference in the training session or grammar checking in the recognition session.

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 311-322

Received 23/11/09
Accepted 16/01/10
Final version 09/03/10

In our method we use structural representation technique. Presented data samples are segmented into simple strokes (primitives) with three properties: *length*, *slope* and *shape (curvature)*. We use a graph to hold of a segmented character. Edges and vertices of the graph represent the strokes and the connection points of these strokes, respectively.

For training and recognition we use the syntactic approach. Segmented and digitally represented shape is converted to an automaton. Conversion process is performed using an intermediate stage where the graph is converted to a sequence of elements, each of which holds the smallest unit of the structure. In the training session, the obtained automaton contributes to the grammar of the specific class and, in the recognition session, is matched to the grammar of each class to detect the best match.

Research for recognition based on curve representation has been reported in the literature. In [12] (topological representation) the primitives of the presented curve are encoded with the slope property and consecutive primitives having the same slope form a part of the curve. The parts are converted into the string representation of the curve (coding). The authors in [2] (coding representation with syntactic recognition) used similar to our's representation except the curvature property. Primitives of a 2D ECG signal were adaptively created using ART2 ANN in [19] (coding representation with syntactic recognition). A series of fuzzy automata, each with a different level of detail, participate in the classification process.

A shape analysis model was reported in [17] (graph representation and template matching recognition). The curves in skeletonized characters are approximated to the sequence of straight line segments and converted into a graph representation. Graphs and trees representation with a mixture of syntactic and template matching recognition was used in [10] to recognize various elements on architectural plans. Straight lines formed the primitive alphabet. Small patterns (windows, doors, stairs, etc.) were represented using graphs and recognition was performed by graph matching. The author in [13] (graphs and trees representation and natural recognition) used SPR to classify objects in medical images. The method learns the structure of the wrist bones with all variations and recognizes defects caused by bone joints or dislocations.

A new method for cursive word recognition was proposed in [18] (coding representation with template matching recognition). Single characters are segmented into straight lines, arcs, ascenders, descenders and loops. Primitives are coded into 1D sequences and the system searches for common rules. Recognition of the traffic signs, reported in [8] (coding representation and neural networks recognition). The author used two types of primitives: lines with the slope property and circles with the breaking point orientation property.

In most of the cases, authors used a small number of properties for segmented curves with complex algorithms for training and recognition. Increasing the complexity of the algorithm will result in an increase in the recognition rates to some extent. Our aim was to develop an approach to handwritten character recognition the main strength of which will be the simplicity and high speed on the

expense of slightly less accurate recognition rates. All main parts of our method are simple and can be implemented with $O(n)$ running time. This method is useful in cases where the computational power of the hardware is not very high and the data to be recognized is not too distorted. As an example we may take pocket computers with handwritten word recognition software plus cooperative user with good handwriting.

The rest of the paper is organized as follows. The lexical analysis and syntactic analysis are explained in detail in Sections 2 and 3, respectively. In Section 4 we explain the practical application of the method proposed. Section 5 concludes the paper.

2 Lexical Analysis

In the lexical analysis phase we prepare the presented data for the syntactic analysis. In this section, we first explain the alphabet and then the segmentation.

2.1 Alphabet

In SPR the alphabet of primitives is obtained either adaptively or is known *a priori*. In our method we decided to use a predefined alphabet of primitives. When the type and format of the data are known, a human supervisor may provide a sufficient alphabet. To represent the strokes of handwritten characters we introduce the term *arc*. An arc is a small segment of a stroke for which the set of property values can be calculated, and it may be completely represented by *length*, *slope* and *shape*, each with a small set of discrete values.

The length of an arc is the number of pixels that form the arc. The assignment of length values may depend on the expected height of the character's body (character without ascenders and descenders).

The slope of an arc is the second property that contains direction information. To determine the slope of an arc, we use a line between the arc ends, called the *chord*. The chord of a sample arc is shown in Fig. 1.

The shape property indicates the direction of the arc's curvature. In calculation of this property we again use the cord. The value of this property may change according to the number, distance and position of each pixel in an arc according to the chord.

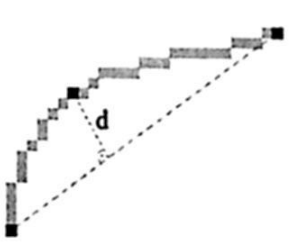

Fig. 1. The farthest point from the chord on the arc.

2.2 Segmentation

Segmentation module is responsible for breaking a given figure into a set of arcs in such a way that all strokes in the figure are encoded by the arcs. In our method, an entire figure is expressed in terms of arcs. In order not to lose the syntactic properties in the figure, arcs have to be connected to each other in the same manner as in the original figure. A good way to represent such a structure is to base the representation on a graph. The edges of the graph are the arcs with the properties converted to numerical values. The vertices of the graph are the common points of the arcs on the original figure.

3 Syntactic Analysis

In the training session, the task of the syntactic component is to generate a grammar for each presented class of data. A grammar is a set of rules that help us distinguish patterns. In the recognition session, the grammar is used as part of input along with the reconstructed signal. Here the task is to determine a *degree of match* with which a new signal matches the grammar presented.

3.1 Conversion from Graph to Sequence of Code Pairs

The graph produced by lexical analysis holds a character in terms of arcs and their interconnections. In the training and recognition sessions we use automata. The structure of the pattern that we use in automata differs from the one produced by the lexical analysis. In order to convert the graph representation of the signal to an automaton we use an intermediate step. In this step, to preserve the 2D topology, a sequence of connected edge pairs, called *code pairs*, is extracted from the initial graph. Each element of the sequence contains two edges of a graph with one common vertex. The length of the sequence obtained from a graph will depend on the number of edges of each vertex. A vertex that connects n edges will produce $C_2^n = \frac{n!}{(n-2)!2!}$ different pairs.

3.2 Grammatical Inference

In the training session, sequences obtained from characters belonging to the same class have to be merged into one learning structure. Knowing that a sequence consists only of primitives and the primitives are unique, we employ automata. An automaton is an appropriate structure for storing syntactic data, assumes that all states are unique, has an ability to learn, and may be trained by infinitely long input sequences.

During the insertion of each code pair to automaton, the state and transition probabilities are updated. In order to always maintain a normalized automaton, we use *linear reward-inaction* (L_{R-I}) probability update method used in learning automata [11]. After the insertion of an arc pair into an automaton, both states (corresponding to codes) are rewarded. To reward more than one state at once we changed the reward and punishment formulae as follows:

$$\begin{aligned} P_i^{t+1} &= P_i^t + (1 - P_i^t)\lambda \frac{1-P_I}{n-P_I} \\ P_k^{t+1} &= P_k^t(1-\lambda) \end{aligned} \tag{1}$$

where n, $0 < \lambda < 1$, P_i, P_k, P_I and t denote the number of the states to be rewarded, learning rate, probability of the rewarded state, the probability of the punished state, the probability sum of the rewarded states, and time, respectively. The last (added) factor in the reward formula distributes total reward according to the prior probabilities of the rewarded states: the higher the probability, the lesser the reward. Transition probabilities are updated using the same formulae where $n = 1$, which neutralizes the last factor.

3.3 Pattern Matching

In the training session we obtain a trained automaton for each class presented. In the recognition session we follow the algorithm explained above and stop after the generation of the sequence of code pairs is completed. Then, instead of generating an automaton, we match the sequence to all trained automata and choose the best match.

When the training session completes, the system ends up with the normalized automata. For the matching operation we still need a few more values.

Matching Parameters During matching, when a sequence of code pairs is presented to a trained automaton, a *score* (τ) is calculated that denotes the presence of the specific arcs and their connections to each other. To calculate the score we search the trained automaton for the states and transitions that represent each pair in the matched sequence:

$$\tau = \sum_{i,j \leq n} P(s_i)P(t_{ij}) + P(s_j)P(t_{ji}) \tag{2}$$

where n is the number of states, $s_i, s_j \in S$ are the present states and $t_{ij}, t_{ji} \in T$ are the transitions between s_i and s_j. In other words, the score shows the extent to which the sequence is present in the automaton.

The automaton of each class produces a score for the sequence of the test character. The scores produced cannot be directly compared. To be able to compare those scores, they have to be converted into a common scale. For this purpose, we use the expected value of the scores that each trained automaton produces for the characters it represents. To determine this value, for each automaton, we use a second pass over the training data to calculate the scores of training characters and the expected values from the collection of obtained scores.

$$E(\tau) = \frac{1}{n} \sum_{i=1}^{n} \tau_i \tag{3}$$

We use this value as a reference point to show that the closer a character's score to the expected value, the higher is the degree of match. The degree of match is calculated by the formula:

$$D_m = 1 - \frac{|E-\tau|}{E^2}\alpha \tag{4}$$

where D_m is the degree of match, E is the expected value (or the average) of the scores, E^2 is a second central moment (or variance), and $\alpha \in (0;1)$ is a constant. To quantify the asymmetry of the distribution of the scores we use the third central moment [15, 16].

Matching In the recognition session, an unknown character passes the lexical analysis phase and is converted to a sequence of code pairs. Next, the sequence obtained is matched against all automata and degrees of match are computed. Finally, the character is assumed to belong to the class whose automaton produced the highest degree of match.

4 Practical Application

In our application we used the Turkish alphabet which contains 29 letters and is similar to the English alphabet. The Turkish alphabet does not use the three letters *q*, *w*, and *x*; instead it includes six new letters *ç*, *ğ*, *ı*, *ö*, *ş*, and *ü*.

4.1 Preprocessing

Before presenting the handwritten character images to the lexical analysis component they have to be preprocessed. The lexical analysis component works with one-pixel thin foreground strokes, so our preprocessing consists of two operations: *binarization* [4] and *skeletonization* [9, 14, 6].

4.2 Segmentation

The first step in segmentation locates the special pixels, called *reference points*, on the skeletonized character. There are two types of such pixels: cross points and ends of the lines.

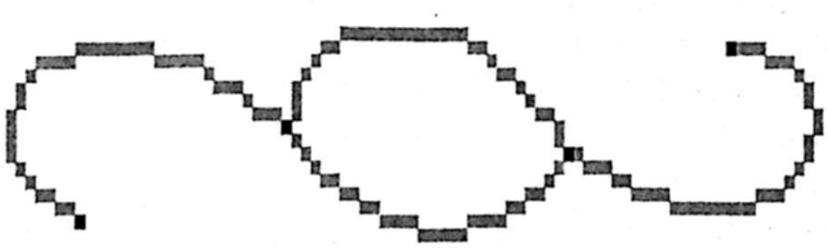

Fig. 2. Cross and end points located on a shape.

A cross point denotes a pixel that has more than two foreground pixels in its 8-pixel neighborhood. An end point is a pixel with only one foreground pixel in its 8-pixel neighborhood. In Fig. 2, the cross points and ends of the lines are determined.

Initially, a stroke between each pair of reference points is assumed to be an arc. The next step is to calculate the values of the properties for each arc. In our application we used a predefined number of values for each property.

The length property may take three values: *short* (s), *medium* (m) and *long* (l).

The slope may take four values: *horizontal* (h), *vertical* (v), *positive* (p) and *negative* (n). If the chord encloses an angle of less than a threshold value (about 15°) with the x-axis or with the y-axis, it is assumed to be horizontal or vertical, respectively. A positive value is assigned in cases when the chord lies in the first and third (positive) quarters of the circle drawn around. The chord locating in the second and fourth (negative) quarters is assigned a negative slope value. For example, the chord of the arc in Fig. 1 is not close to any of the axes and lies in the positive quarters of the circle, so the slope property value of that arc is positive.

The shape property may take three values: *straight* (s), *convex* (x) and *concave* (v). To determine the shape property value we draw a chord and calculate the average distance d_{avg} of the pixels in the arc to the chord and the number of pixels, p_a and p_b, vertically above and below the chord, respectively. The average distance will show us the degree of the arc's concavity and the number of points will help us distinguish between convex and concave values. The arc is assumed to be straight if the degree of concavity is less than a specific threshold value. If the arc is not straight, we check the number of pixels above and below the chord. If $p_a > p_b$ the shape property of the arc is set to convex, and in the case of $p_a < p_b$, to concave.

There are $3 \times 4 \times 3 = 36$ different value combinations for an arc. Each combination is called the *code*. Codes are generated as follows: each property value is given a number that is unique for each property (length: $s = 1$, $m = 2$, $l = 3$; slope: $h = 1$, $v = 2$, $p = 3$, $n = 4$; shape: $s = 1$, $x = 2$, $v = 3$). The code is a number that contains all properties in the same order: $code = 100 \times length + 10 \times slope + 1 \times shape$. For example, an arc with property values as medium, positive and convex is encoded to $100 \times 2 + 10 \times 3 + 1 \times 2 = 232$.

The properties of these arcs may fail to be calculated: an arc may be too long or short, or may be a too complex curve to be clearly defined as convex or concave. Such arcs are broken up into several simpler arcs by locating additional reference points. The leftmost and the rightmost arcs in Fig. 2 are examples of complex arcs. To locate additional reference points we use the *bounding box operation*, where an arc is surrounded by a tight (bounding) box. The pixels that touch the sides of the box become new reference points. In Fig. 3 two points for the leftmost and the rightmost arcs were located using this operation.

In the case that the bounding box operation does not produce any new reference point, we apply the *farthest point operation*, where the pixel that has

the highest distance from the chord, as shown in Fig. 1, is the additional reference point. In Fig. 3 one point for the middle arcs were located using the farthest point operation.

The segmentation process continues until all arcs are assigned valid values for their three properties.

The representation of the segmented character is based on the reconstruction of the original figure in terms of primitives (arc codes) instead of arcs and presenting it as a graph. For instance, the graph corresponding to the symbol in Fig. 3 consists of 10 vertices and edges, as shown in Fig. 4. We assume that the code of the arc contains sufficient information about the arc and we do not have to keep the pixel details.

Fig. 3. Property values' abbreviations and codes of the all arcs in the shape.

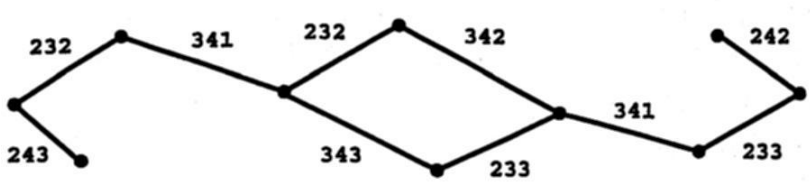

Fig. 4. Graph representation of the shape in Fig. 3.

4.3 Code Pairs

To convert graph obtained from the lexical analysis to the sequence of code pairs we extract all possible combinations of connected edges from each vertex. As states earlier the number of pairs from a vertex that connects n edges will be $C_2^n = \frac{n!}{(n-2)!2!}$. The only exception is the vertex that contains one edge, which produces the pair with 0 on one side. For instance, the graph in Fig. 4 produces the following sequence: 0-243, 243-232, 232-341, 341-232, 341-343, 232-343, 232-342, 343-233, 342-233, 342-341, 233-341, 341-233, 233-242 and 242-0. The order of the pairs is irrelevant since each pair is a part of the graph topology and the order of the codes in a pair is also irrelevant since the edges in a graph are not directed.

4.4 Automaton

Presenting one element of a sequence, a code pair, to an automaton consists of a few simple and straightforward operations. First, each code in a code pair is assumed to represent a state of an automaton. New states are created if the automaton does not contain a newly presented code. Second, a connection between the codes in the pair represents a transition between states. We create two transitions with opposite directions for each code pair [20], because transitions in an automaton are directed. The automaton corresponding to the graph in Fig. 4 is shown in Fig. 5.

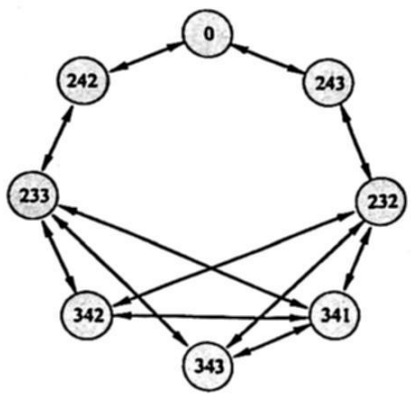

Fig. 5. Automaton representation of the graph in Fig. 4.

4.5 Results And Discussion

The main strength of our method is its simplicity and time complexity. All main parts of our method are simple and can be implemented with $O(n)$ running time. In the training session, the algorithm requires two passes over the training data. The runtime of the second pass can be made negligible by using only a small portion of the training data. For example, if an automaton was built using 10000 samples, then the second pass may be completed with only 100 samples chosen randomly. Segmentation is performed by a single pass over all pixels of a sample image and all arcs extracted are used once in an automaton generation. All structures used in our method are simple and require limited amount of memory. Each trained automaton consists of 36 states and 36×36 transitions at most.

Since there is no database available for Turkish handwritten character recognition, we compiled a new database. The data set of 12000 handwritten characters was collected from 10 writers. After testing our system on the Turkish handwritten characters, we obtained promising results in the area of character recognition. Each character in the test set was presented to all automata and degrees of match provided by each automaton were sorted in descending order. In 71.94% of the cases the score of the correct automaton was the highest (positioned on the top of the order) and in 93.79% of the cases was among the top

five results. Distribution of the recognition rates from the top result to the top five results is shown in Fig. 6.

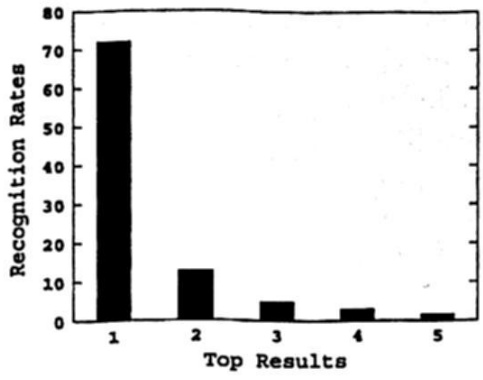

Fig. 6. Average positions of the correct automata in the recognition order.

The major part of the correct recognitions is concentrated on a first position. The positions not shown in figure have the average values less than one percent.

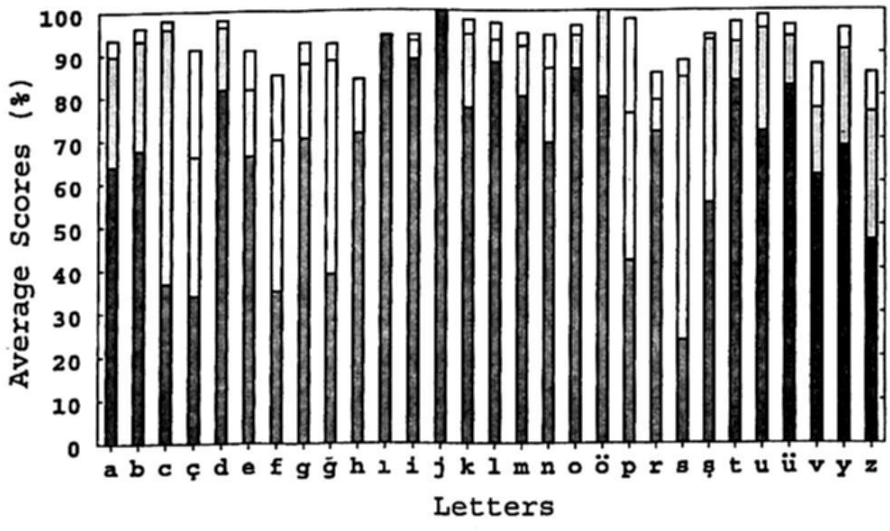

Fig. 7. Distribution of the top five results for each letter.

In our system, the simpler the topology of a character, the higher the recognition performance. The distribution of the recognition results for each character is shown in Fig. 7. The top result is marked with the darkest portion of the bars. The top second and third results are combined and marked with the gray color. The fourth with fifth results are also combined and shown with the white colored portions of the bars. As we expected, the simplest letters such as *ı*, *i*, *j*, *l*, *o*, *ö* and *t* have top recognition rates of more than 80%, while the lowest rates were calculated for the letters *c*, *ç*, *f*, *ğ* and *s* with many curved portions.

We also studied the recognition errors for some of the most frequent letters in Turkish: *a*, *e*, *i*, *l*, *n* and *r* (Table 1). Most frequently *a* is erroneously recognized

Table 1. Misclassifications of some of the most frequent letters in Turkish. The tables show the erroneous choices and their percentages among all misclassifications of the letter in the header of a relevant table.

a	e	i	l	n	r
o 35.96%	v 23,95%	l 26,53%	i 57,14%	i 21,85%	i 39,80%
ü 15.79%	r 20,96%	r 22,45%	r 19,05%	ş 17,65%	ş 18,37%
e 15.35%	o 11,38%	ü 12,24%	v 7,14%	r 16,81%	l 13,27%
c 7.02%	i 10,78%	ş 10,20%	ç 4,76%	e 12,61%	c 5,10%
u 5.70%	c 10,18%	v 8,16%	ş 4,76%	ü 11,76%	e 5,10%

as *o*, *ü*, *e*, *c* and *u*. The potential reason is that those letters have a similar topology with *a*. Misclassifications for the same reason may be observed for *i*, *l* and *r*. The letter *e* is mostly misclassified with *v* and *r*, because they have a similar handwritten topology. The misclassifications of *n* do not display a proper pattern, because *n* can be easily distorted during handwriting, and hence, may have a similar topology with many letters.

5 Conclusion

In this paper we proposed a constructive learning automaton approach to the recognition of handwritten characters. Promising results were obtained after testing our system. Recognition rates reached vary between 71.94% for the best and 93.79% for the top five results. For unconstrained characters segmented from words, performance of our approach is similar to those reported in [21, 22]. A similar approach was used in [3]. The authors reached an average top five recognition rate of 93.78% for online English handwritten character recognition.

As can be concluded from the results the weakest point in our application is the representation technique of smooth curves that are usually encountered in letters *c*, *ç*, *g*, *ğ*, *s* and *ş*. The direction of possible future work is the revision of the representation technique for smooth curves. A new technique for placement of reference points may be focused on the sharp changes of the degree of curvature along the presented curve. This will lead to robustness against slant variation and increase the performance.

Acknowledgements. This work has been supported by the Boğaziçi University Research Fund under the grant number 09A107D.

References

1. N. Arica and F.T. Yarman-Vural. An overview of character recognition focused on off-line handwriting. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions*, 31:216–233, 2001.
2. Y. Assabie and J. Bigun. Ethiopic character recognition using direction field tensor. *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, 3:284–287, 2006.

3. V. S. Chakravarthy and B. Kompella. The shape of handwritten characters. *Pattern Recognition Letters*, 24:1901–1913, 2003.
4. D.A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach.* Prentice Hall, 2003.
5. R. Gonzalez and M. Thomason. *Syntactic Pattern Recognition: An Introduction.* Addison-Wesley, 1978.
6. C.M. Holt, A. Stewart, M. Clint, and R.H. Perrott. An improved parallel thinning algorithm. *Communications of the ACM*, 30(2):156–160, 1987.
7. K.-Y. Huang. *Syntactic Pattern Recognition For Seismic Oil Exploration.* World Scientific, 2002.
8. S. Kantawong. Road traffic signs detection and classification for blind man navigation system. *Control, Automation and Systems, 2007. ICCAS '07. International Conference on*, pages 847–852, 2007.
9. L. Lam, S.-W. Lee, and C.Y. Suen. Thinning methodologies - a comprehensive survey. *Pattern Analysis and Machine Intelligence*, 14(9):869–885, 1992.
10. J. Llados and G. Sanchez. Symbol recognition using graphs. *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, 2:49–52, 2003.
11. K. Narendra and M. Thathatchar. *Learning Automata: An Introduction.* New York. Addison-Wesley, 1989.
12. H. Nishida and S. Mori. Algebraic description of curve structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:516–533, 1992.
13. M.R. Ogiela. Automatic understanding of medical images based on grammar approach. *Imaging Systems and Techniques, 2007. IST '07. IEEE International Workshop on*, pages 1–4, 2007.
14. B.R. Okombi-Diba, J. Miyamichi, and K. Shoji. Segmentation of spatially variant image textures. *16th International Conference on Pattern Recognition (ICPR'02)*, 2:20917, 2002.
15. A. Papoulis. *Probability, Random Variables and Stochastic Processes.* McGraw-Hill Companies; 3rd edition, 1991.
16. W. Paul and J. Baschnagel. *Stochastic Processes: From Physics to Finance.* Springer, 1999.
17. J. Rocha and T. Pavlidis. A shape analysis model with applications to a character recognition system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:393–404, 1994.
18. J.C. Simon. Off-line cursive word recognition. *Proceedings of the IEEE*, 80:1150–1161, 1992.
19. M.B. Tümer, L.A. Belfore, and K.M. Ropella. A syntactic methodology for automatic diagnosis by analysis of continuous time measurements using hierarchical signal representations. *Systems, Man, and Cybernetics, Part B, IEEE Transactions on*, 33(33):951–965, 2003.
20. A. Ustimov and B. Tümer. Construction of a learning automaton for cycle detection in noisy data sequences. *Lecture Notes in Computer Science*, pages 543–552, 2005.
21. B.A. Yanikoglu and P.A. Sandon. Off-line cursive handwriting recognition using style parameters. *Department of Mathematics and Computer Science, Dartmouth College*, 1993.
22. B.A. Yanikoglu and P.A. Sandon. Recognizing off-line cursive handwriting. *Proc. Computer Vision and Pattern Recognition*, pages 397–403, 1994.

Utterances Assessment in Chat Conversations

Mihai Dascalu[1,2], Stefan Trausan-Matu[1,3], Philippe Dessus[4]

[1] University "Politehnica" of Bucharest,
313, Splaiul Indepentei, 060042 Bucharest, ROMANIA

[2] S.C. CCT S.R.L.,
30, Gh Bratianu, 011413 Bucharest, ROMANIA

[3] Romanian Academy Research Institute for Artificial Intelligence,
13, Calea 13 Septembrie, Bucharest, ROMANIA

[4] Grenoble University,
1251, av. Centrale, BP 47, F-38040 Grenoble CEDEX 9, FRANCE
mikedascalu@yahoo.com, stefan.trausan@cs.pub.ro, Philippe.Dessus@upmf-grenoble.fr

Abstract. With the continuous evolution of collaborative environments, the needs of automatic analyses and assessment of participants in instant messenger conferences (chat) have become essential. For these aims, on one hand, a series of factors based on natural language processing (including lexical analysis and Latent Semantic Analysis) and data-mining have been taken into consideration. On the other hand, in order to thoroughly assess participants, measures as Page's essay grading, readability and social networks analysis metrics were computed. The weights of each factor in the overall grading system are optimized using a genetic algorithm whose entries are provided by a perceptron in order to ensure numerical stability. A gold standard has been used for evaluating the system's performance.

Keywords: assessment of collaboration, analysis of discourse in conversation, social networks, LSA, Computer Supported Collaborative Learning.

1 Introduction

As a result of the ongoing evolution of the web, new collaboration tools emerged and with them the desire to thoroughly process large amounts of information automatically. From the Computer Supported Collaborative Learning's (CSCL) point of view [1], chats play an important role and have become more and more used in the effective learning process. On the other hand, manual assessment of chats is a time consuming process from the teacher's side, and therefore the need to develop applications that can aid the evaluation process has become essential. From this perspective the major improvement targeted by this paper is the development of an automatic assessment system in order to evaluate each participant in a chat environment. A series of natural language processing and social network analysis methods were used, in addition with other computed metrics for assessment.

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 323-334

Received 24/11/09
Accepted 16/01/10
Final version 09/03/10

The system was used for CSCL chats in which teams of 4-8 students were asked to discuss, without a moderator, the benefits of online collaboration tools. Each of the students was assigned to support a collaborative technology (wikis, blogs, chats and forums), arguing both pros and cons for it. The language was English and the environment used was Concert Chat [6], which offers the possibility of explicit referencing previous utterances. From the obtained corpus, 80 chats were afterwards manually evaluated by a student from a different year for not influencing the assessment process.

The next section of this paper will present the metrics used in the evaluation process starting from the simplest, as readability or Page's factors, initially used for essay grading [3], moving to social network analysis and finally *Latent Semantic Analysis* (LSA) for a semantic approach of the marking system. The third section evaluates the system.

2 The Evaluation Process

Communication between participants in a chat is conveyed through language in a written form. Lexical, syntactic, and semantic information are the three levels used to describe the features of written utterances [2], and will be taken into account for the analysis of a participant's involvement in a chat. First, *surface metrics* are computed for all the utterances of a participant in order to determine factors like fluency, spelling, diction or utterance structure [2, 3]. All these factors are combined and a mark is obtained for each participant without taking into consideration a lexical or a semantic analysis of what they are actually discussing. At the same level *readability* ease measures are computed.

The next step is *grammatical and morphological analysis* based on spellchecking, stemming, tokenization and part of speech tagging. Eventually, a *semantic evaluation* is performed using LSA [4]. For assessing the on-topic grade of each utterance a set of predefined keywords for all corpus chats is taken into consideration.

Moreover, at the surface and at the semantic levels, metrics specific to social networks are applied for proper assessment of participants' involvement and similarities with the overall chat and predefined topics of the discussion.

2.1 Surface Analysis

In order to perform a detailed surface analysis two categories of factors are taken into consideration at a lexical level: Page's essay grading proxes and readability. Page's idea was that computers could be used to automatically evaluate and grade student essays as effective as any human teacher using only simple measures – statistically and easily detectable attributes [5]. The main purpose was to prove that computers could grade as well, but with *less effort and time*, therefore enabling teachers to *assign more writing*. So the goal was to improve the student's capabilities by practice, having at hand the statistical capabilities of computers for writing analysis.

In order to perform a statistical analysis, Page correlated two concepts: *proxes* (computer approximations of interest) with human *trins* (intrinsic variables – human

measures used for evaluation). The overall results were remarkable – a correlation of 0.71 using only simple measures which proved that computer programs could predict grades quite reliably - at least the grades given by the computer correlated with the human judges as well as the humans had correlated with each other.

Starting for Page's metrics [5] for automatically grading essays, and taking into consideration Slotnick's method [5] to group them correspondingly to their intrinsic values, the following factors and values were identified in order to evaluate each participant only at the surface level:

Table 1. Categories taken into consideration and corresponding proxes

Number	Quality	Characteristic Proxes
1.	Fluency	Number of total characters, number of total words, number of different words, mean number of characters per utterance, number of utterances, number of sentences (different, because in an utterance multiple sentences can be identified)
2.	Spelling	Misspelled words, but in order to obtain a positive approach (the greater the percentage, the better) the percentage of correctly written words is used
3.	Diction	Mean and standard deviation of word length
4.	Utterance Structure	Number of utterances, mean utterance length in words, mean utterance length in characters

All the above proxes determine the average consistency of utterances. Although simple, all these factors play an important role in discovering the most important person in a chat, in other words to measure his activity. In addition, quantity is also important in its part of analyzing each participant's utterances.

Each factor has the same weight in the corresponding quality and the overall grade is obtained by using the arithmetic mean on all predefined values. All these factors, except misspelled words, are converted into percentages in order to scale them and to obtain a relative mark for all participants.

The second factor taken into account is readability. It can be defined as *reading ease* of a particular text, especially as it results from one's writing style. This factor is very important because extensive research in this field show that easy-reading text (and in our case chats and utterances) has a great impact on comprehension, retention, reading speed, and reading persistence.

Because readability implies the interaction between a participant and the collaborative environment, several features from the *reader's point of view* are essential: prior knowledge, personal skills and traits (for example intelligence), interest, and motivation.

In the currently evaluated chats, the first factor (prior knowledge) can be considered approximately the same for all students because all come from the same educational environment and share a common background. On the other hand, the remaining features vary greatly from one student to another and the last two ones are greatly reflected in their implication in the chat.

Therefore two key aspects must be taken into consideration: *involvement* and *competency*, both evaluated from the social network's point of view and with a semantic approach which will be detailed further in this paper.

Starting from Jacques Barzun's quote –"Simple English is no person's native tongue"– it is very difficult to write for a class of readers other than one's own, therefore readability plays an important role in understanding a chat. Although in a chat environment some words are omitted and syntax is usually simplified, readability still offers a good perspective of one's current level of knowledge/understanding or attitude in some cases, but all the information obtained from readability measures must be correlated with other factors.

Readability is commonly used unconsciously, based on the insight of other chat participants, but for its evaluation a readability formula is used, which is calibrated against a more labor-intensive readability survey and which matches the overall text with the expected reading level of the audience [4]. These formulas estimate the reading skill required to read the utterances in a chat and evaluate the overall complexity of the words used, therefore providing the means to target an audience.

Three formulas were computed. *The Flesch Reading Ease Readability Formula* (http://www.readabilityformulas.com/flesch-reading-ease-readability-formula.php) is one of the oldest and most accurate readability formulas, providing a simple approach to assess the grade-level of a chat participant and the difficulty of reading the current text. This score rates all utterances of a user on a 100 point scale. The higher the score, the easier it is to read, not necessarily understand the text. A score of 60 to 70 is considered to be optimal.

$$RE = 206{,}835 - (1{,}015 * ASL) - (84{,}6 * ASW). \quad (1)$$

RE is the **R**eadability **E**ase, **ASL** is the **A**verage **S**entence **L**ength (the number of words divided by the number of sentences) and **ASW** is the **A**verage number of **S**yllables per **W**ord (he number of syllables divided by the number of words).

The *Gunning's Fog Index* (or FOG) *Readability Formula* (http://www.readabilityformulas.com/gunning-fog-readability-formula.php) is based on Robert Gunning's opinion that newspapers and business documents were full of "fog" and unnecessary complexity. The index indicates the number of years of formal education a reader of average intelligence would need to understand the text on the first reading. A drawback of the Fog Index is that not all multi-syllabic words are difficult, but for computational issues, the consideration that all words above 2 syllables are complex is used.

$$FOG = (ASL + PHW) * 0{,}4. \quad (2)$$

ASL is the **A**verage **S**entence **L**ength (the number of words divided by the number of sentences) and **PHW** is the **P**ercentage of **H**ard **W**ords (in current implementation words with more than 2 syllables and not containing a dash)

The Flesch Grade Level Readability Formula (http://www.readabilityformulas.com/flesch-grade-level-readability-formula.php) rates utterances on U.S. grade school level. So a score of 8.0 means that the document can be understood by an eighth grader. This score makes it easier to judge the readability level of various texts in order to assign them to students. Also, a document

whose score is between 7.0 and 8.0 is considered to be optimal, since it will be highly readable.

$$FKRA = (0{,}39 * ASL) + (11{,}8 * ASW) - 15{,}59 \quad (3)$$

FKRA is the **F**lesch-**K**incaid **R**eading **A**ge, **ASL** is the **A**verage **S**entence **L**ength (the number of words divided by the number of sentences) and **ASW** is the **A**verage number of **S**yllable per **W**ord (the number of syllables divided by the number of words)

For each given chat, the system performs and evaluates all the 3 previous formulas and provides to the user detailed information for each participant. Also relative correlations between these factors and the manual annotation grades are computed in order to evaluate their relevance related to the overall grading process.

2.2 Social Networks Analysis

In addition to quantity and quality measures computed starting from the utterances, social factors are also taken into account in our approach. Consequently, a graph is generated from the chat transcript in concordance with the utterances exchanged by the participants. Nodes are participants in a collaborative environment and ties are generated based on explicit links (obtained from the explicit referencing facility of the chat environment used [6], which enables participants to manually add links during the conversation for marking subsequent utterances derived from a specific one).

From the point of view of social networks, *various metrics* are computed in order to determine the most competitive participant in chat: degree (indegree, outdegree), centrality (closeness centrality, graph centrality, eigen–values) and user ranking similar to the well known *Google Page Rank Algorithm* [7]. These metrics are applied first on the effective number of interchanged utterances between participants providing a quantitative approach; Second, the metrics are applied to the sum of utterance marks based on a semantic evaluation of each utterance; the evaluation process will be discussed in section 2.5 and, based on the results obtained for each utterance, a new graph is built on which all social metrics are applied. This provides the basis for a qualitative evaluation of the chat.

All the identified metrics used in the social network analysis are *relative* in the sense they provide markings relevant only compared with other participants in the same chat, not with those from other chats. This is the main reason why all factors are scaled between all the participants, giving each participant a weighted percentage from the overall performance of all participants.

2.3 LSA and the Corresponding Learning Process

Latent Semantic Analysis is a technique based on the *vector-space based model* [10, 14]. It is used for analyzing relationships between a set of documents and terms contained within by projecting them in sets of concepts related to those documents [9, 10]. LSA starts from a *term-document array* which describes the occurrence of each term in all the corpus documents. LSA transforms the occurrence matrix into a

relation between terms and concepts, and a relation between those concepts and the corresponding documents. Thus, the terms and the documents are now indirectly related through concepts [10, 13]. This transformation is obtained by a singular-value decomposition of the matrix and a reduction of its dimensionality.

Our system uses words from a chat corpus. The first step in the learning process, after spell–checking, is stop words elimination (very frequent and irrelevant words like "the", "a", "an", "to", etc.) from each utterance. The next step is POS tagging and, in case of verbs, these are stemmed in order to decrease the number of corresponding forms identified in chats by keeping track of only the verb's stem (the meaning of all forms is actually the same, but in LSA only one form is learnt). All other words are left in their identified forms, adding corresponding tagging because same words, but with different POS tags have other contextual senses, and therefore semantic neighbors [11].

Once the term-document matrix is populated, *Tf-Idf* (term frequency - inverse document frequency [13]) is computed. The final steps are the singular value decomposition (SVD) and the projection of the array in order to reduce its dimensions. According to [12], the optimal empiric value for k is 300, a value used in current experiments at which multiple sources concord.

Another important aspect in the LSA learning process is segmentation which is the process of dividing chats taking into consideration units with similar meaning and high internal cohesion. In the current implementation, the chat is divided between participants because of the considered unity and cohesion between utterances from the same participant. These documents are afterwards divided into segments using fixed non-overlapping windows. In this case contiguous segments are less effective because of intertwined themes present in chats and these aspects will be dealt with in future improvements of the marking system.

LSA is used for evaluating the proximity between two words by the *cosine measure*:

$$Sim(word_1, word_2) = \frac{\sum_{i=1}^{k} word_{1,i}, word_{2,i}}{\sqrt{\sum_{i=1}^{k} word_{1,i}^2} \times \sqrt{\sum_{i=1}^{k} word_{2,i}^2}} \cdot \quad (4)$$

Similarities between utterances and similarities of utterances related with the entire document are used in order to assess the importance of each utterance compared with the entire chat or with a predefined set of keywords referenced as a new document:

$$Vector(utterance) = \sum_{i=1} (1 + \log(no\_occurence(word_i)) * vector(word_i) \cdot \quad (5)$$

$$Sim(utterance_1, utterance_2) = Sim(Vector(utterance_1), Vector(utterance_2)) \cdot \quad (6)$$

2.4 The Utterance and Participants' Evaluation Process

2.4.1 The Utterance Marking Process

The first aspect that needs to be taken care of is building the graph of utterances which highlights the correlations between utterances on the basis of explicit references.

In order to evaluate each sentence, after finishing the morphological and lexical analysis three steps are processed:

1. Evaluate each utterance individually taking into consideration several features: the effective length of initial utterance; the number of occurrences of all keywords which remain after eliminating stop words, spell-checking and stemming; the level at which the current utterance is situated in the overall thread (similar to a Breadth-First search in the utterance space/threads based only on explicit links); the branching factor corresponding with the actual number of derived utterances from current one; the correlation / similarity with the overall chat; the correlation / similaritude with a set of predefined set of topics of discussion.

This mark combines the quantitative approach (the length of the sentence starting from the assumption that a piece of information should be more valuable if transmitted in multiple messages, linked together, and expressed in more words, not only to impress, but also meaningful in the context) with a qualitative one (the use of LSA and keywords).

In the process of evaluating each utterance, the semantic value is evaluated with the help of likelihood between the terms used in the current utterance (those after preliminary processing) and the whole document, respectively those from a list of predefined topics of discussion.

The formulas used for evaluating each utterance are:

$$mark_{empiric} = \left(\frac{length(initial\_utterance)}{10} + \frac{9}{10} \times \sum_{word}^{remaining} mark(word) \right) \times . \\ \times emphasis \tag{7}$$

$$mark(word) = length(word) * (1 + \log(no\_occurences)) \; . \tag{8}$$

$$emphasis = (1 + \log(level)) \times (1 + \log(branching\_factor) \times \\ \times Sim(utterance, whole\_document) \times \\ \times Sim(utterance, predefined\_keywords) \tag{9}$$

2. Emphasize Utterance Marks. Each thread obtained by chaining utterances based upon explicit links has a global maximum around which all utterance marks are increased correspondingly with a Gaussian distribution:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{(x-\mu)^2}{2\sigma^2}}, \text{ where:} \quad (10)$$

$$\sigma = \frac{\max(id\_utter\_thread) - \min(id\_utter\_thread)}{2}; \quad (11)$$

$$\mu = id\_utterance\_with\_highest\_mark\ . \quad (12)$$

Therefore each utterance mark is multiplied by a factor of $1 + p(currrent\_utterance)$.

3. *Determine the final grade for each utterance in the current thread*

Based upon the empiric mark, the final mark of the utterance is obtained for each utterance in its corresponding thread:

$$mark_{final} = mark_{final}(prev\_utter) + coefficient \times mark_{empiric}\ , \quad (13)$$

where the coefficient is determined from the type of the current utterance and the one to which it is tied to.

For the coefficient determination, identification of speech acts plays an important role: verbs, punctuation signs and certain keywords are inspected. Starting from a set of predefined types of speech acts, the coefficients are obtained from a predefined matrix. These predefined values were determined after analyzing and estimating the impact of the current utterance considering only the previous one in the thread (similar to a Markov process). The grade of a discussion thread may be raised or lowered by each utterance. Therefore, depending on the type of an utterance and the identified speech acts, the final mark might have a positive or negative value.

2.4.2 Participant Grading

The in-degree, out-degree, closeness and graph centrality, eigen–values and rank factors are applied on the matrix with the number of interchanged utterances between participants and the matrix which takes into consideration the empiric mark of an utterance instead of the default value of 1. Therefore, in the second approach quality, not quantity is important (an element $[i, j]$ equals the sum of $mark_{empiric}$ for each utterance from participant i to participant j), providing a deeper analysis of chats using a social network's approach based on a semantic utterance evaluation.

Each of the analysis factors (applied on both matrixes) is converted to a percentage (current grade/sum of all grades for each factor, except the case of eigen centrality where the conversion is made automatically by multiplying with 100 the corresponding eigen–value in absolute value). The final grade takes into consideration all these factors (including those from the surface analysis) and their corresponding weights:

$$final\_grade_i = \sum_k weight_k \times percentage_{k,i}\ , \quad (24)$$

where k is a factor used in the final evaluation of the participant i and the weight of each factor is read from a configuration file.

After all measures are computed and using the grades from human evaluators, the Pearson correlation for each factor is determined, providing the means to assess the importance and the relevance compared with the manual grades taken as reference.

General information about the chat – for example overall grade correlation, absolute and relative correctness – are also determined and displayed by the system.

2.5 Optimizing each Metric's Grade

The scope of the designed algorithm is to determine the optimal weights for each given factor in order to have the highest correlation with the manual annotator grades. A *series of constraints* had to be applied. First, *minimal/maximum values* for each weight are considered. For example, a minimum of 2% in order to take into consideration at least a small part of each factor, and maximum 40% in order to give all factors a chance and not simply obtain a solution with all factors 0% besides the one with the best overall correlation – 100%. Second, *the Sum of all factors* must be 100%. Third, obtain *maximum mean correlation* for all chats in the corpus.

In this case, the system has two components. A *perceptron* is used for obtaining fast solutions as inputs for the genetic algorithm. The main advantages for using this kind of network are the capacity to learn and adapt from examples, the fast convergence, the numerical stability; search in the weight space for optimal solution; duality and correlation between inputs and weights.

Secondly, a *genetic algorithm* is used for fine-tuning the solutions given by the neural network, also keeping in mind the predefined constraints. This algorithm operates over a population of chromosomes which represent potential solutions. Each generation represents and approximation of the solution - the determination of optimal weights in order to assure the best overall correlation, not the best distance between automatic grades and annotator ones. Correlation is expressed as an arithmetic mean of all correlations per chat because of the differences between evaluator styles.

The scope of this algorithm is to **maximize** the **overall correlation**, and specific characteristics of the implemented algorithm are:

- ***Initialization***: 2/3 of initial population obtained via Neural Networks (perceptron), the rest is randomly generated in order to avoid local;
- ***Fixed number of*** 100 chromosomes per population;
- ***Fitness*** - overall correlation of all chats from the corpus evaluated as a mean of all individual correlations;
- ***Selection*** – roulette based or elitist selection - the higher the fitness, the greater the possibility a participant is selected for crossover;
- ***Correction*** – a necessary operator in order to assure that the initial constraint are satisfied: if above or below minim/maximum values, reinitialize weight starting from threshold and adding a random quantity to it; if overall sum of percentages different from 100% adjust randomly weights with steps of 1/precision;
- ***Crossover*** - is based on *Real Intermediate Recombination* which has the highest dispersion of newly generated weights - select a random alpha for

each factor between [-0,25; 1,25]; the relative distance between 2 chromosomes selected for crossover must be at least 20% in order to apply the operator over them;

- Use ***CHC optimization***, with a little modification - generate N children and retain 20% of the best newly generated chromosomes; 20% of best parents are kept in the new generation and the rest is made of the best remaining individuals;
- ***Multiple populations*** that exchange best individuals - add after 10 generations the best individual to a common list and replace the worst individual with a randomly selected one from the list;
- After reaching ***convergence*** of a population (consecutively 20% of the maximum number of generations have the same best individual), reinitialize population = keep best 10% of existing individuals, obtain 30% via neural networks, and generate the remaining randomly;

The solution for determining the optimal weights combines the two approaches in order to obtain benefits from both – numerical stable solutions from neural networks and the flexibility of genetic algorithms in adjusting these partial solutions.

3 System Evaluation

The initial running configuration used by the system was: 10% for Page's Grading, 5% for social networks factors applied on the number of interchanged utterances, and 10% for the semantic social network factors applied on utterance marks. The overall results obtained with these weights are: *Relative correctness*: 77.44%, *Absolute correctness*: 70.07%, Correlation: 0.514.

Relative correctness and absolute correctness represent absolute/relative distances in a one-dimensional space, where the annotator's grade and the one obtained automatically using the Ch.A.M.P. system are taken into consideration for the given corpus. Eventually, the final results (as arithmetic means for each of the 3 individual measures determined per chat) are also displayed.

The results after multiple runs of the weight optimization system (all with 4 concurrent populations) show that most importance in the manual evaluation process is given to the following factors:

Table 2. Results after multiple runs of the weight optimization system, with regards to factors with a corresponding percentage $\geq 10\%$

Percentage	Factor
20-25%	Page's Grading methods - so only surface analysis factors
10-15%	Indegree from the social network's point of view, applied on number of interchanged utterances
30-40%	Outdegree also determined by the number of outgoing utterances – somehow a participant's gregariousness measure
$\approx$ 10%	Semantic graph centrality – the only measure with a higher importance applied which relies on utterance marks

All remaining factor are evaluated below 5%, therefore don't have high importance in the final grading process. The overall results, with regards to correlation optimization, obtained after running the genetic algorithm are: *Relative correctness*: ≈ 46.83%, *Absolute correctness*: ≈ 45.70%, *Correlation*: ≈ 0.594.

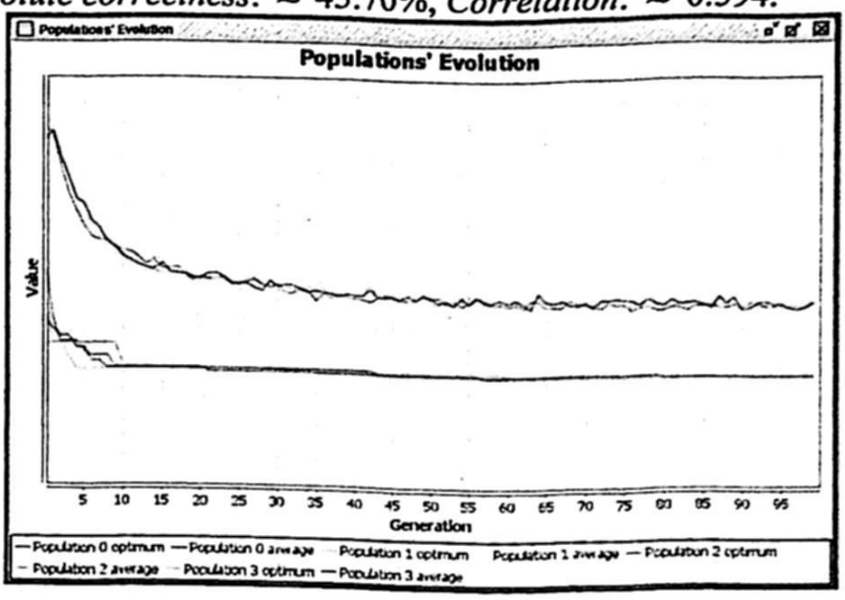

Fig. 1. Convergence to an optimal solution using 4 populations with the visualization of optimum/average chromosomes

The spikes from each population's average fitness are determined by newly inserted individuals or by the population reinitialization. After the first 10 iterations important improvements can be observed, whereas after 30 generations the optimum chromosomes of each population stagnate. Only population reinitializations and chromosome interchanges provide minor improvements in the current solution.

Our results entail several conclusions: The human grading process uses a predominantly quantitative approach; Uncorrelated evaluations and different styles/principles used by different human annotators are the main causes for lowering the overall correlation and correctness; The improvement of correlation was in the detriment of absolute/relative correctness; Convergence of the genetic algorithm can be considered after 30 generations.

4 Conclusions

The results obtained from our system allow us to conclude that the evaluation of a participant's overall contribution in a chat environment can be achieved. Also we strongly believe that with further tuning of the weights, better LSA learning and increased number of social network factors (including those applied to the entire network) will increase performance and reliability of the results obtained. Moreover, the subjective factor in manual evaluation is also present and influences the overall correctness.

In present, evaluations and tuning of the assessment system are performed in the LTfLL project, in which the work presented in the paper is one of the modules for feedback generation [16].

Acknowledgements

The research presented in this paper was partially performed under the FP7 EU STREP project LTfLL. We would like to thank all the students of University "Politehnica" of Bucharest, Computer Science Department, who participated in our experiments and provided the inputs for generating our golden standard.

References

1. Stahl, G.: Group Cognition: Computer Support for Building Collaborative Knowledge. MIT Press (2006)
2. Anderson, J. R.: Cognitive psychology and its implications, New York, Freeman (1985)
3. Page, E. B. Paulus, D. H.: Analysis of essays by computer. Predicting Overall Quality, U.S. Department of Health, Education and Welfare (1968)
4. http://www.streetdirectory.com/travel_guide/15672/writing/all_about_readability_formulas__and_why_writers_need_to_use_them.html
5. Wresch, W.: The Imminence of Grading Essays by Computer--25 Years Later. Computers and Composition 10(2), 45-58, retrieved from http:// computersandcomposition.osu.edu/archives/v10/10_2_html/10_2_5_Wresch.html (1993)
6. Holmer, T., Kienle, A. & Wessner, M.: Explicit Referencing in Learning Chats: Needs and Acceptance. In: Nejdl, W., Tochtermann, K., (eds.): Innovative Approaches for Learning and Knowledge Sharing- ECTEL, LNCS, 4227, Springer, pp. 170–184 (2006)
7. Dascălu, M., Chioaşcă, E.-V. Trăuşan-Matu, S.: ASAP – An Advanced System for assessing chat participants. In: D. Dochev, M. Pistore, and P. Traverso (Eds.): AIMSA 2008, LNAI 5253, Springer, pp. 58–68 (2008)
8. Bakhtin, M. M.: Problems of Dostoevsky's poetics (Edited and translated by Caryl Emerson). Minneapolis: University of Minnesota Press (1993)
9. http://lsa.colorado.edu/
10. Landauer, K. Th., Foltz, W. P., Laham, D.: An Introduction to Latent Semantic Analysis. Discourse Processes, 25, 259-284 (1998)
11. Wiemer-Hastings, P., Zipitria, I.: Rules for syntax, vectors for semantics. In: proceeding of the 23rd Annual Conference of the Cognitive Science Society (2001).
12. Lemaire, B.: Limites de la lemmatisation pour l'extraction de significations. JADT 2008: 9es Journées internationales d'Analyse statistique des Données Textuelles (2008).
13. Manning, C., Schütze, H.: Foundations of statistical Natural Language Processing. MIT Press, Cambridge (Mass.) (1999)
14. Miller, T.: Latent semantic analysis and the construction of coherent extracts. In: Nicolov, N. Botcheva, K., Angelova, G. and Mitkov, R., (eds.), Recent Advances in Natural Language Processing III. John Benjamins, Amsterdam/Philadelphia, pp. 277–286 (2004)
15. Fernandez, S., Velazquez, P., Mandin, S.: Les systèmes de résumé automatique sont-ils vraiment des mauvais élèves?. In: JADT 2008: 9es Journées internationales d'Analyse Statistique des Données Textuelles (2008)
16. Stefan Trausan-Matu, Traian Rebedea, A Polyphonic Model and System for Inter-animation Analysis in Chat Conversations with Multiple Participants, in A. Gelbukh (Ed.): CICLing 2010, LNCS 6008, Springer, 2010, pp. 354–363

Punctuation Detection with Full Syntactic Parsing

Miloš Jakubíček and Aleš Horák

Faculty of Informatics
Masaryk University
Botanická 68a, 602 00 Brno
Czech Republic
xjakub@fi.muni.cz, hales@fi.muni.cz

Abstract. The correct placement of punctuation characters is in many languages, including Czech, driven by complex guidelines. Although those guidelines use information of morphology, syntax and semantics, state-of-art systems for punctuation detection and correction are limited to simple rule-based backbones. In this paper we present a syntax-based approach by utilizing the Czech parser **synt**. This parser uses an adapted chart parsing technique for building the chart structure for the sentence. synt can then process the chart and provide several kinds of output information. The implemented punctuation detection technique utilizes the synt output in the form of automatic and unambiguous extraction of optimal syntactic structures from the sentence (noun phrases, verb phrases, clauses, relative clauses or inserted clauses). Using this feature it is possible to obtain information about syntactic structures related to expected punctuation placement. We also present experiments proving that this method makes it possible to cover most syntactic phenomena needed for punctuation detection or correction.

1 Introduction

Incorrect usage of punctuation characters such as comma, semi-colon, or dash has always been a common mistake in written texts, especially in languages with such complex (and strict) guidelines for punctuation placement as in the case of the Czech language [1]. It has been shown that mistakes in punctuation (21 % of all errors) represent the second most frequent category of mistakes in Czech – the first position is occupied by stylistics with 23 % [2].

It is no wonder that automatic detection and correction of punctuation for Czech is still an open task and the state-of-art systems usually lack both precision and recall around 50 % [3], as they use only a small-to-medium set of rules matching the simplest guidelines for punctuation. It is obvious that such methods do not cover phenomena on syntactic or semantic levels and that such approach cannot be easily adapted to do so.

In this paper we focus on superseding the deficiencies of current punctuation detection systems with analysis on the syntactic level by using **synt**, a powerful

Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 335-343

Received 27/11/09
Accepted 16/01/10
Final version 10/03/10

and feature-rich syntactic parser for Czech (more on the parser in Section 2). The main reason of the technique is that since the parse is able to parse (i. e. *recognize*) punctuation in the input, it also might be able to fill in (i. e. *generate*) missing punctuation. We show that with relatively small set of post-processing rules this method achieves significantly better results than the current systems (as described in [3]).

The synt parser provides several options of output information based on the packed *chart structure* containing the resulting analyses. Besides standard enumeration of all phrasal or dependency trees, synt can compute the optimal decomposition of the input text to selected syntactic structures such as noun, verb or prepositional phrases, clauses etc. The extraction of structures (described in details in Section 3) allows us to obtain all the necessary syntactic information needed for filling in the punctuation into the given sentence.

The structure of this papers is as follows: in Sections 2 and 3 we briefly describe the synt parser and the extraction of structures, then in Section 4 we explain how we adapt it and use it for punctuation detection and finally we present results of our work in Section 5.

1.1 Related Work

Since the Czech rules for placing punctuation are so complicated, this topic has been addressed by several authors, however, with partial success only. There are two commercially available products which try to tackle this task *Grammaticon* [4] and *Grammar Checker* [5], which is included in the Czech localisation of MS Word text editor. A comparison of both systems has been made by Pala in 2005 [3] showing that both of them lack especially recall. A proof-of-concept system has been also shown in [6] trying to use Inductive Logic Programming to solve this task.

The main problem of current solutions is that they are designed as rather simple rule-based systems with a set of (hard-coded or inducted, context-free or context-sensitive) rules trying to describe the placement of Czech punctuation. Although many of the principles for placing punctuation have syntactic background, none of them applies full syntactic parsing for this task. In this paper we utilize the Czech parse synt and show that a syntax-based approach has promising results.

2 The synt Parser

The Czech parser synt [7,8] has been developed in the Natural Language Processing Centre at Masaryk University. It performs a head-driven chart-type syntactic analysis based on the provided context-free grammar with additional contextual tests and actions. For easy maintenance, this grammar is edited in the form of a metagrammar (having about 200 rules) from which the full grammar can be automatically derived (having almost 4,000 rules). The per-rule defined

contextual actions are used to cover phenomena such as case-number-gender agreement.

In recent measures [9, p. 77] it has been shown that synt accomplishes a very good coverage (above 90 %) but the analysis is highly ambiguous: for some sentences even millions of output syntactic trees can occur. There are several strategies developed to cope with the ambiguity: first, the grammar rules are divided into different priority levels which are used to prune the resulting set of output trees. Second, every resulting chart edge has a ranking value assigned from which the ranking for the whole tree can be efficiently computed in order to sort the trees and output N best trees while keeping it in polynomial time.

The synt parser contains (besides the chart parsing algorithm itself) many additional functions such as maximum optimal coverage of partial analyses (shallow parsing) [10], effective selection of n-best output trees [11] or chart and trees beautification [12]. The punctuation detection technique uses the function of extraction of syntactic structures [13], which is described in detail in the next section.

3 Extraction of Phrase Structures

Usually, a derivation tree is presented as the main output of syntactic parsing of natural languages, but currently most of the syntactic analysers for Czech lack precision. However there are many situations in which it is not necessary and sometimes even not desirable to require such derivation trees as the output of syntactic parsing, may it be simple information extraction and retrieval, transformation of sentences into a predicate-arguments structure or any other case, in which we rather need to process whole syntactic structures in the given sentence, especially noun, prepositional and verb phrases, numerals or clauses. Moreover, so as not to deal with the same problems as with the tree parser output, we need to identify these structures unambiguously.

The phrase extraction functionality in synt enables us to obtain syntactic structures from the analysed sentence that correspond to a given grammar nonterminal in a number of ways. For the purpose of phrase extraction, the internal parsing structure of synt is used, the so called *chart*, a multigraph which is built up during the analysis holding all the resulting trees. An important feature of chart is its polynomial size [7, p. 133] implying that it is a structure suitable for further effective processing[1] – as the number of output trees can be exponential to the length of the input sentence, processing of each tree separately would be otherwise computationally infeasible.

However, as the algorithm works directly with chart and not trees, it is very fast and can be used for processing massive amount of data in a short time, even if we are extracting many structures at once (see Table 1 for details).

The output of the extraction is shown in the following two examples of extracting clauses:

[1] By processing the chart we refer to the result of the syntactic analysis, i.e. to the state of the chart after the analysis.

- **Example 1.**
  Input:
  Muž, který stojí u cesty, vede kolo.
  (A man who stands at the road leads a bike.)
  Output:
  `[0-9): Muž , , vede kolo`
  (a man leads a bike)
  `[2-6): který stojí u cesty`
  (who stands at the road)
- **Example 2.**
  Input:
  Vidím ženu, která drží růži, jež je červená.
  (I see a woman who holds a flower which is red.)
  Output:
  `[0-3): Vidím ženu ,`
  (I see a woman)
  `[3-7): která drží růži ,`
  (who holds a flower)
  `[7-10): jež je červená`
  (which is red)

4 Punctuation Detection

The main idea of using the syntactic parser with extraction of structures for punctuation detection is as follows: if the parser has anyway to expect (i. e. match and parse) the actual presence of the punctuation using its grammar rules, we may try to modify those grammar rules in such a way that enables[2] the punctuation detection even if the punctuation is (by mistake) not present in the sentence, and then extract the related (empty) punctuation nonterminals as described in

[2] by allowing empty productions in the grammar

Table 1. Results of detection on a sample set of 500 sentences.[†]

Step	Precision	Recall
Adding ϵ-rules	35.37 %	20.12 %
Further grammar modifications	69.43 %	57.31 %
Matching coordinations	**82.27 %**	**84.76 %**
Sentences	500	
Average time needed per sentence	0.65 s	

[†] The precision and recall were measured across the whole sentence set.

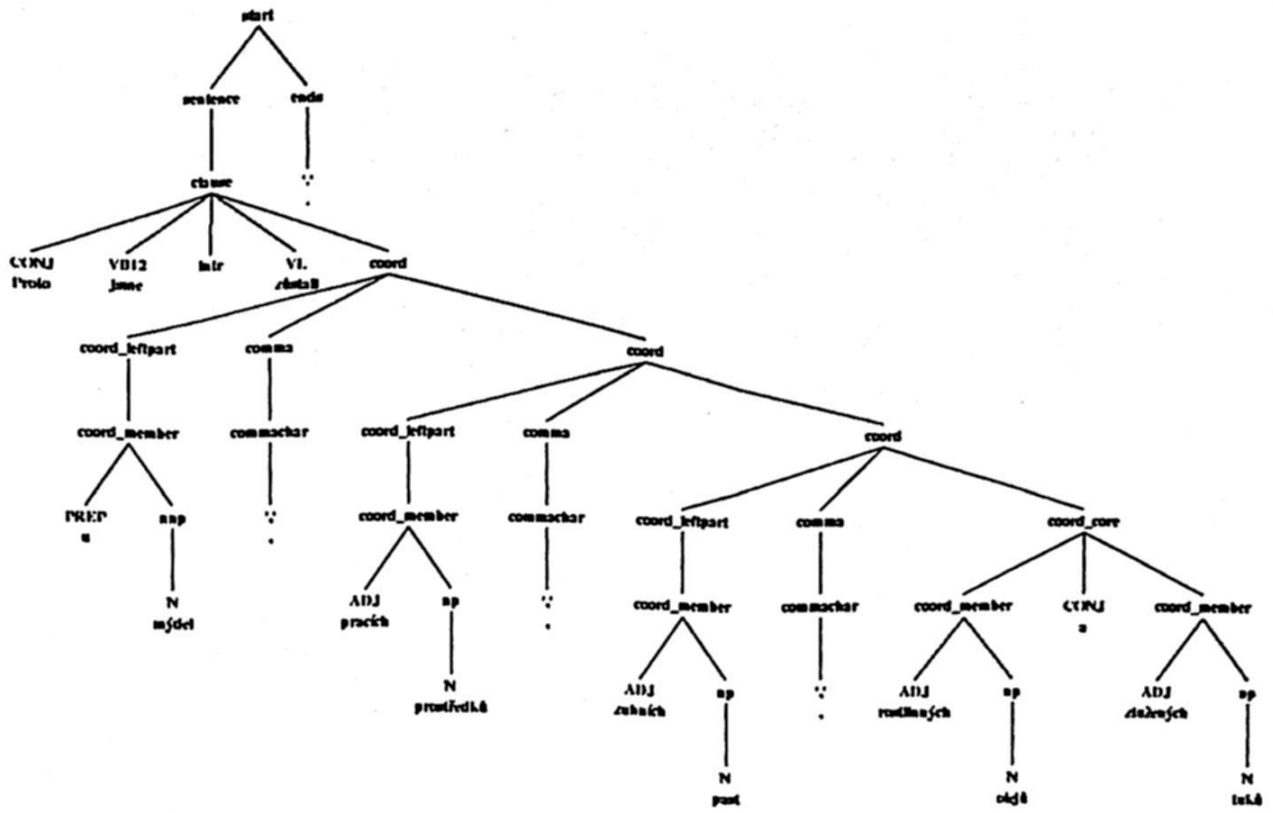

Fig. 1. Example tree for an input sentence *Proto jsme zůstali u mýdel, pracích prostředků, zubních past, rostlinných olejů a ztužených tuků* (Thus we continued with soaps, detergents, tooth pastes, vegetable oils and hardened fats).

the previous section. Since the extraction of the punctuation basically represents a projection from the chart structure to the sentence surface structure, the missing punctuation can be identified as a post-analysis step.

The difference between parsing a present punctuation mark and detecting a missing punctuation is displayed in Figures 1 and 2: while the first one represents a regular derivation tree for a sentence containing punctuation, the other one shows an almost identical derivation tree that, however, was produced from a sentence in which no punctuation (except the trailing full stop) was present, but the parser still deduced the correct placement of comma nonterminals. The whole process of punctuation detection is then demonstrated in Figure 3.

As the first step, we have modified the relevant grammar punctuation rules to allow empty productions.[3] Even this trivial change in the grammar led to a recall of $< 20\,\%$ (see Table 1 for details), therefore we further analysed the grammar rules and improved them in order to better fit the purpose of punctuation detection.

The grammar modifications mainly focused on improving the ability to parse (and hence also detect) punctuation in common Czech sentences (especially be-

[3] I. e. for each rule covering some punctuation marks, e. g. comma → ",", we added a rule allowing the empty production: comma → ϵ.

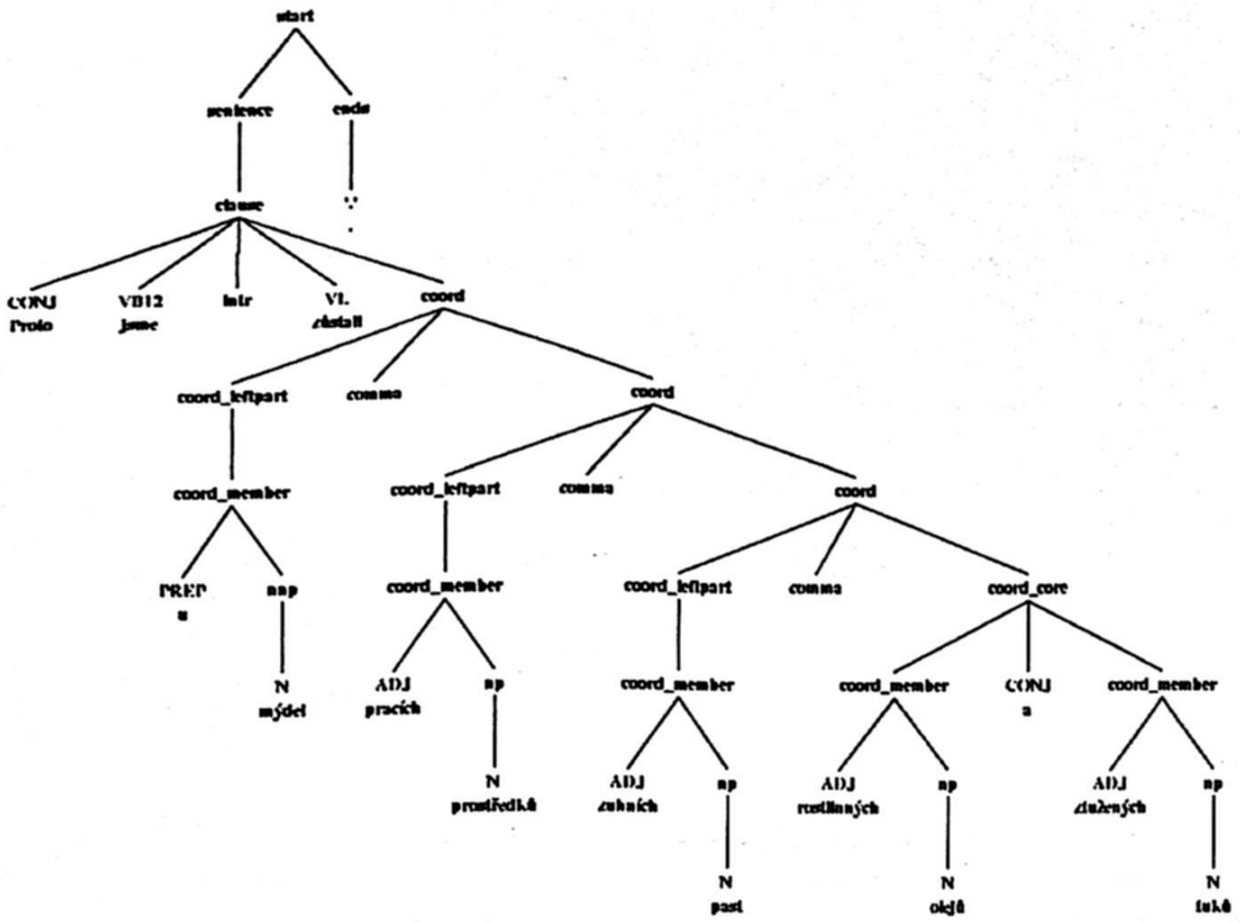

Fig. 2. Example tree for the same sentence from which punctuation has been removed.

tween clauses, relative clauses and conjunctions), the recall increased to almost 60 %.

Finally we paid special attention to coordinations of phrases where punctuation plays an important role.[4] Detecting correct placement of punctuation marks in coordinations required new grammar rules to distinguish coordinations (in which all members have to agree in grammatical case) from common noun or prepositional phrases (where no such grammatical restrictions are applied) as shown in the example derivation tree with coordinations (Figure 2). The resulting recall as well as precision increased to more than 80 %.

4.1 Evaluation and Results

For the purpose of evaluation of the described method, 500 randomly chosen sentences from the manually annotated DESAM corpus [14] have been used. Firstly, we removed all punctuation marks from the given sentence, then we filled in the punctuation using the enhanced parser and finally we compared

[4] In general, the punctuation here distinguishes the meaning of the coordinations. This however requires semantic information (see Section 4.2), therefore we concentrated to syntactic phenomena in coordinations.

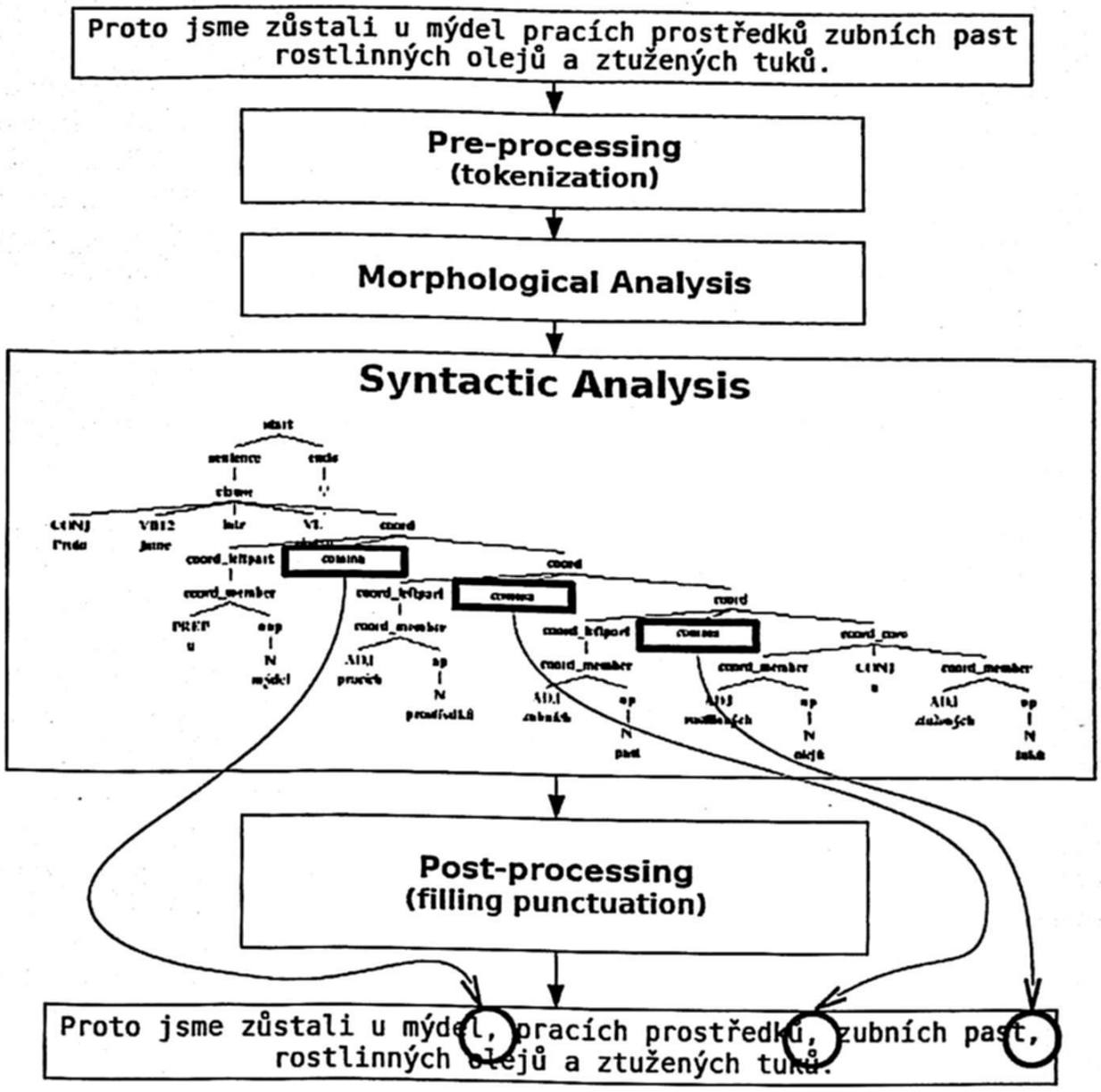

Fig. 3. Overview of the punctuation detection process.

it with the original sentence. The presented results confirm the importance of coordinations for solving this problem, because they contain many punctuation characters as is indicated by the increase in recall by almost 30 %. Also, as can be seen from the value of precision, the coordinations can be precisely analysed on the syntactic level.

4.2 Problems and Limitations

As mentioned above, the guidelines for placing punctuation in Czech take information not only from morphology and syntax, but also from semantics – such phenomena cannot be covered by a pure syntax-based system. Using semantic information for punctuation detection is necessary in order to distinguish e. g.:

- *coordinations from gradually extending attributes*
  In Czech, coordinations are written with punctuation, but gradually extending attributes without it, e.g. *Je to velký starý drahý dům* (It is a big old expensive house), but *Vidím velký, střední a malý dům* (I see a large, medium and small house) Currently, we consider a noun phrase sequence to be a coordination only if it contains a coordination core as its part, consisting from a phrase followed by a conjunction and another phrase[5], i.e. *střední a malý* (medium and small) in the previous sentence. However, the coordination core does not need to be present in some (stylistically determined) situations.
- *sequences of subordinating clauses*
  If considering the sentence *Petr si uvědomil, že Pavel zavřel okno[,] a šel domů* (Peter realized that Paul closed the window[,] and went home), the presence of the comma determines whether it was Peter or Paul who went home.

Obviously, for proper detection of comma placement in the first example, it would be necessary to have the knowledge of the meaning of the sentences or at least of some parts of the sentence – moreover, we would need to know the relation between the meanings: to know that *large, medium, small* belong to the same semantic class while *big, old, expensive* does not. It turns out that to tackle the problem, large ontologies for Czech would be needed which are unfortunately not available.

The situation in the second example is even worse: we would need large context, anaphora resolution and logical inference to deduce the proper placement of the punctuation and even then, the situation might be just ambiguous from the available information.

5 Conclusions and Future Directions

In this paper we have presented an efficient method for punctuation detection in common sentences by employing full syntactic parsing. Although all measures were related to Czech, we believe that the technique might be easily generalized for other languages with complicated punctuation rules provided that they have the necessary language resources and tools.

The proposed method has already achieved significantly better results than the state-of-art systems and we believe that it might be further improved by supplying additional information which could overcome its current limitations described in the previous section. In particular, we plan to use the Czech WordNet [15] to distinguish coordinations and gradually extending attributes, and search for other semantic resources that may help us by improving this method.

Furthermore, an application consisting of an OpenOffice.org [16] grammar checker extension is being developed for practical testing of the punctuation detection technique and for making it available to a broad scope of users.

[5] Both phrases must also agree in case.

Acknowledgments

This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 and in the National Research Programme II project 2C06009 and by the Czech Science Foundation under the project P401/10/0792.

References

1. Hlavsa, Z., et al.: Akademická pravidla českého pravopisu (Rules of Czech Orthography). Akademia, Praha (1993)
2. Pala, K., Rychlý, P., Smrž, P.: Text corpus with errors. In: Text, Speech and Dialogue: Sixth International Conference, Berlin, Springer Verlag (2003) 90–97
3. Pala, K.: Pište dopisy konečně bez chyb – Český gramatický korektor pro Microsoft Office. Computer **13–14** (2005) 72 CPress Media.
4. Lingea s. r. o.: Grammaticon. `http://www.lingea.cz/grammaticon.htm` (2003)
5. Oliva, K., Petkevič, V., Microsoft s.r.o.: Czech grammatical checker. `http://office.microsoft.com/word` (2005)
6. Pala, K., Nepil, M.: Checking punctuation errors in czech texts. [online, quoted on 2009-11-20]. Available from: `http://nlp.fi.muni.cz/publications/neco2003\_pala\_nepil/neco2003\_pala\_nepil.rtf` (2003)
7. Horák, A.: The Normal Translation Algorithm in Transparent Intensional Logic for Czech. PhD thesis, Faculty of Informatics, Masaryk University, Brno (November 2001)
8. Kadlec, V., Horák, A.: New Meta-grammar Constructs in Czech Language Parser synt. In: Lecture Notes in Computer Science, Springer Berlin / Heidelberg (2005)
9. Kadlec, V.: Syntactic Analysis of Natural Languages Based on Context-Free Grammar Backbone. PhD thesis, Faculty of Informatics, Masaryk University, Brno (September 2007)
10. Ailomaa, M., Kadlec, V., Rajman, M., Chappelier, J.C.: Robust stochastic parsing: Comparing and combining two approaches for processing extra-grammatical sentences. In Werner, S., ed.: Proceedings of the 15th NODALIDA Conference, Joensuu 2005, Joensuu, Ling@JoY (2005) 1–7
11. Kovář, V., Horák, A., Kadlec, V.: New Methods for Pruning and Ordering of Syntax Parsing Trees. In Proceedings of Text, Speech and Dialogue 2008. In: Lecture Notes in Artificial Intelligence, Proceedings of Text, Speech and Dialogue 2008, Brno, Czech Republic, Springer-Verlag (2008) 125–131
12. Kovář, V., Horák, A.: Reducing the Number of Resulting Parsing Trees for the Czech Language Using the Beautified Chart Method. In: Proceedings of 3rd Language and Technology Conference, Poznań, Wydawnictwo Poznańskie (2007) 433–437
13. Jakubíček, M.: Extraction of syntactic structures based on the czech parser synt. In: Proceedings of Recent Advances in Slavonic Natural Language Processing 2008, Brno, Czech Republic, Masaryk University (2008) 56–62
14. Pala, K., Rychlý, P., Smrž, P.: DESAM – Annotated Corpus for Czech. In: Proceedings of SOFSEM '97, Springer-Verlag (1997) 523–530
15. Pala, K., Hlaváčková, D.: Derivational Relations in Czech WordNet. In: Proceedings of the Workshop on Balto-Slavonic Natural Language Processing 2007, Praha, Czech Republic, The Association for Computational Linguistics (2007) 75–81
16. OpenOffice.org Community: Openoffice.org. `http://www.openoffice.org` (2009)

Author Index

Editorial Board of the Volume

Eneko Agirre
Sivaji Bandyopadhyay
Roberto Basili
Christian Boitet
Nicoletta Calzolari
Dan Cristea
Alexander Gelbukh
Gregory Grefenstette
Eva Hajičová
Yasunari Harada
Graeme Hirst
Eduard Hovy
Nancy Ide
Diana Inkpen
Alma Kharrat
Adam Kilgarri
Igor Mel'čuk
Rada Mihalcea
Ruslan Mitkov
Dunja Mladeníc
Masaki Murata
Vivi Nastase
Nicolas Nicolov
Kemal Oflazer
Constantin Orasan
Maria Teresa Pazienza
Ted Pedersen
Viktor Pekar
Anselmo Peñas
Stelios Piperidis
James Pustejovsky
Fuji Ren
Fabio Rinaldi
Roracio Rodriguez
Vasile Rus
Franco Salvetti
Serge Sharo
Grigori Sidorov
Thamar Solorio
Juan Manuel Torres-Moreno
Hans Uszkoreit
Manuel Vilares Ferro
Leo Wanner
Yorick Wilks
Annie Zaenen

Additional Referees

Rodrigo Agerri
Muath Alzghool
Javier Artiles
Bernd Bohnet
Ondřej Bojar
Nadjet Bouayad-Agha
Luka Bradesko
Janez Brank
Julian Brooke
Miranda Chong
Silviu Cucerzan
Lorand Dali
Víctor Manuel Darriba Bilbao
Amitava Das
Dipankar Das
Arantza Díaz de Ilarraza
Kohji Dohsaka
Iustin Dornescu
Asif Ekbal
Santiago Fernández Lanza
Robert Foster
Oana Frunza
René Arnulfo García Hernández
Ana García-Serrano
Byron Georgantopoulos
Chikara Hashimoto
Laura Hasler
William Headden
Maria Husarciuc
Adrian Iftene
Iustina Ilisei
Ikumi Imani
Aminul Islam
Toshiyuki Kanamaru

Fazel Keshtkar
Jason Kessler
Michael Kohlhase
Olga Kolesnikova
Natalia Konstantinova
Valia Kordoni
Hans-Ulrich Krieger
Geert-Jan Kruij
Yulia Ledeneva
Yang Liu
Oier Lopez de Lacalle
Fernando Magán-Muñoz
Aurora Marsye
Kazuyuki Matsumoto
Alex Moruz
Sudip Kumar Naskar
Peyman Nojoumian
Blaz Novak
Inna Novalija
Tomoko Ohkuma
Bostjan Pajntar
Partha Pakray
Pavel Pecina
Ionut Cristian Pistol
Natalia Ponomareva
Marius Raschip
Luz Rello
Francisco Ribadas
German Rigau
Alvaro Rodrigo
Franco Salvetti
Kepa Sarasola
GeroldSchneider
Marc Schroeder
Ivonne Skalban
Simon Smith
Mohammad G. Sorba
Tadej Štajner
Sanja Štajner
Jan Strakova
Xiao Sun
Masanori Suzuki
Motoyuki Suzuki
Motoyuki Takaai
Irina Temnikova
Zhi Teng
Nenad Tomasev
Eiji Tomida
Sulema Torres
Mitra Trampas
Diana Trandabat
Stephen Tratz
Yasushi Tsubota
Hiroshi Umemoto
Masao Utiyama
Andrea Varga
Tong Wang
Ye Wu
Keiji Yasuda
Zhang Yi
Daisuke Yokomori
Caixia Yuan
Zdeně Zabokrtský
Venta Zapirain
Daniel Zeman
Hendrik Zender

Natural Language Processing is an interdisciplinary research area at the border between linguistics and artificial intelligence aiming at developing computer programs capable of human-like activities related to understanding or producing texts or speech in a natural language, such as English or Chinese.

This volume includes 27 original research papers by authors from 25 different countries on the following areas of theory and applications of natural language processing:

- Semantics
- Morphology, Syntax, Named Entity Recognition
- Opinion, Emotions, Textual Entailment
- Text and Speech Generation
- Machine Translation
- Information Retrieval and Text Clustering
- Educational Applications
- Applications

The volume is oriented to researchers and students working in natural language processing, computational linguistics, and human language technologies, as well as to all readers interested in these areas.

ISSN: 1870-4069
www.ipn.mx
www.cic.ipn.mx

SEP

INSTITUTO POLITÉCNICO NACIONAL
"La Técnica al Servicio de la Patria"