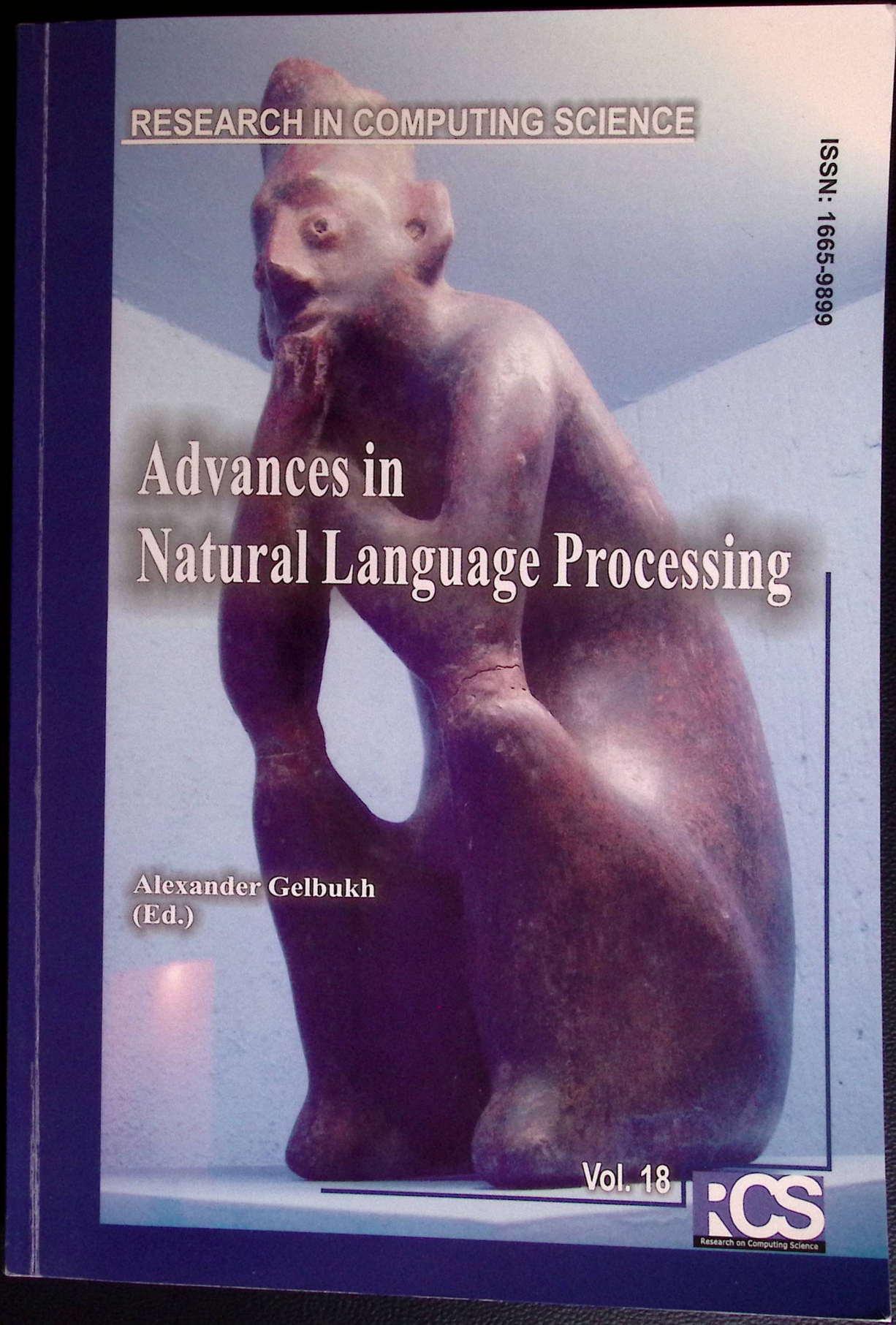

RESEARCH IN COMPUTING SCIENCE
ISSN: 1665-9899
Advances in
Natural Language Processing
Alexander Gelbukh
(Ed.)
Vol. 18
RCS
Research on Computing Science

Advances in Natural Language Processing

Research in Computing Science

Series Editorial Board

Comité Editorial de la Serie

Editors-in-Chief:

Editores en Jefe

Juan Humberto Sossa Azuela (Mexico)
Gerhard Ritter (USA)
Jean Serra (France)
Ulises Cortés (Spain)

Associate Editors:

Editores Asociados

Jesús Angulo (France)
Jihad El-Sana (Israel)
Jesús Figueroa (Mexico)
Alexander Gelbukh (Russia)
Ioannis Kakadiaris (USA)
Serguei Levachkine (Russia)
Petros Maragos (Greece)
Julian Padget (UK)
Mateo Valero (Spain)

Editorial Coordination:

Coordinación Editorial

Blanca Miranda Valencia

Formatting:

Formación

Sulema Torres Ramos

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 18**, Febrero, 2006. 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. Imagen de la portada: www.absolutewallpapers.com/wallpapers/3dwallpapers/fractal/fractal_8.jpg. 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*

***Research in Computing Science** is published by the Center for Computing Research of IPN. **Volume 18**, February, 2006. Printing 500. The authors are responsible for the contents of their articles. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior permission of Centre for Computing Research. Printed in Mexico City, February, 2006, in the IPN Graphic Workshop – Publication Office.*

Volume 18

Volumen 18

Advances in Natural Language Processing

Volume Editor:
Editor del Volumen

Alexander Gelbukh

Instituto Politécnico Nacional
Centro de Investigación en Computación
México 2006







ISSN: 1665-9899

Copyright © Instituto Politécnico Nacional 2006
Copyright © by Instituto Politécnico Nacional

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

Printing: 500
Impresiones: 500

Printed in Mexico
Impreso en México

Preface

Natural Language Processing is a branch of Artificial Intelligence aimed at enabling the computers to perform meaningful human-like activities related to written or spoken human language, such as English or Spanish. For traditional computer programs, a text is just a long string of characters—which the program can copy, print, or erase. In contrast, intelligent natural language processing programs are designed for operations involving the meaning of the text.

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 synthesis 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 19 original research papers written by 63 authors representing 16 different countries: Argentina, Austria, Canada, China, Finland, Hong Kong, India, Japan, Korea, Lebanon, Mexico, Norway, Spain, UK, United Arab Emirates, and USA. The volume is structured in 7 thematic areas of both theory and applications of Natural Language Processing:

- Lexical Resources
- Morphology and syntax
- Word Sense Disambiguation and Semantics
- Speech Processing
- Information Retrieval
- Question Answering and Text Summarization
- Computer-Assisted Language Learning

The papers included in this volume were selected on the base of rigorous international reviewing process out of 43 submissions received for evaluation; thus the acceptance rate of this volume was 44%.

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 giving sense to the work of all other people involved, as well 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, Ignacio Garcia Araoz, Oralia del Carmen Pérez Orozco, and Raquel López Alamilla 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

February 2006

Table of Contents
Índice

Page/Pág.

Lexical Resources

Generating Multilingual Grammars from OWL Ontologies 3
Guillermo Pérez, Gabriel Amores, Pilar Manchón and David González
Clustering of English-Korean Translation Word Pairs Using Bi-grams 15
Hanmin Jung, Hee-Kwan Koo, Won-Kyung Sung and Dong-In Park

Morphology and syntax

Modified Makagonov's Method for Testing Word Similarity and its Application to Constructing Word Frequency List 27
Xavier Blanco, Mikhail Alexandrov, Alexander Gelbukh
HMM based POS Tagger for a Relatively Free Word Order Language 37
Arulmozhi Palanisamy and Sobha Lalitha Devi
Extended Tagging in Requirements Engineering 49
Günther Fliedl, Christian Kop, Heinrich C. Mayr, Jürgen Vöhringer, Georg Weber and Christian Winkler
Analogical Modeling with Bias allowing Feedback and Centering 57
Christer Johansson and Lars G. Johnsen

Word Sense Disambiguation and Semantics

Word Sense Disambiguation with Basic-Level Categories 71
Steve Legrand
A New Proposal of Word Sense Disambiguation for Nouns on a Question Answering System 83
S. Ferrández, S. Roger, A. Ferrández, A. Aguilar and P. López-Moreno
On Types, Intension and Compositionality 93
Walid S. Saba

Speech Processing

Acoustic Model Adaptation for Codec Speech based on Learning-by-Doing Concept 105
Shingo Kuroiwa, Satoru Tsuge, Koji Tanaka, Kazuma Hara and Fuji Ren
Specific Speaker's Japanese Speech Corpus over Longand Short Time Periods 115
Satoru Tsuge, Masami Shishibori, Fuji Ren, Kenji Kita and Shingo Kuroiwa
A Task Analysis of Nursing Activites Using Spoken Corpora 125
Hiromi itoh Ozaku, Akinori Abe, Kaoru Sagara, Noriaki Kuwahara and Kiyoshi Kogure

Information Retrieval

Improved Focused Web Crawling Based on Incremental Q-Learning 139
Yunming Ye, Yan Li, Joshua Huang and Xiaofei Xu
Automatic Identification of Chinese Stop Words 151
Feng Zou, Fu Lee Wang, Xiaotie Deng and Song Han
Fast Search Algorithm for Sequences of Hierarchically Structured Data 163
Masaki Murata, Masao Utiyama, Toshiyuki Kanamaru and Hitoshi Isahara

Question Answering and Text Summarization

Cross-Lingual Question Answering Using InterLingual Index Module of EuroWordNet 177
Sergio Ferrández and Antonio Ferrández
Using Terminology and a Concept Hierarchy for Restricted-Domain Question-Answering 183
Hai Doan-Nguyen and Leila Kosseim
Efficient Randomized Algorithms for Text Summarization 195
Ahmed Mohamed and Sanguthevar Rajasekaran

Computer-Assisted Language Learning

Arabic Error Feedback in an Online Arabic Learning System 203
Khaled F. Shaalan and Habib E. Talhami

Author Index 213

Editorial Board of the Volume 215

Additional Reviewers 215

Lexical Resources

Generating Multilingual Grammars from OWL Ontologies

Guillermo Pérez, Gabriel Amores, Pilar Manchón, and David González

Universidad de Sevilla
Seville, Spain
{gperez, jgabriel, pmanchon, dgmaline}@us.es

Abstract. This paper describes a tool which automatically generates productions for context-free grammars from OWL ontologies, using just a rule-based configuration file. This tool has been implemented within the framework of a dialogue system, and has achieved several goals: it leverages the manual work of the linguist, and ensures coherence and completeness between the Domain Knowledge (Knowledge Manager Module) and the Linguistic Knowledge (Natural Language Understanding Module) in the application.

1 Introduction

1.1 Automatic Grammar Generation

The problem of manually generating grammars for a Natural Language Understanding (NLU) system has been widely discussed. Two main approaches can be highlighted from those proposed in the literature: Grammatical Inference and Rule Based Grammar Generation.

The Grammatical Inference approach (http://eurise.univ-st-etienne.fr/gi/) refers to the process of learning grammars and languages from data and is considered nowadays as an independent research area within Machine Learning techniques. Examples of applications based on this approach are ABL [1] and EMILE [2].

On the other hand, the Rule Based approach tries to generate the grammar rules from scratch (i.e. based on the expertise of a linguist), while trying to minimize the manual work. An example of this approach is the Grammatical Framework [3], whose proposal is to organize the multilingual grammar construction in two building blocks: an abstract syntax which contains category and function declarations, and a concrete syntax which defines linearization rules. Category and function declarations are shared by all languages and thus appear only once, while linearization rules are defined on a per-language basis. Methods which generate grammars from ontologies (including ours) may also be considered examples of the Rule Based approach.

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 3-14

1.2 Generating Grammars from Ontologies

In the context of Dialogue Management, the separation of the Knowledge Manager (the module in charge of the domain knowledge) and the NLU module poses a number of advantages: the complexity of the linguistic components is reduced [4], the existing domain knowledge may be reused [4], reference resolution processes (i.e: anaphoric resolution, underspecification, presupposition, and quantification) are better defined [5] , and, finally, it helps the dialogue manager in keeping dialogue coherence [4].

However, the information contained in the Knowledge Manager module overlaps with information inside the NLU module. The key idea of this paper is that this redundancy can be used to automatically generate grammar rules from the relationships between the concepts described in the ontology. Thus, the fact that the concept "lamp" is linked to the concept "blue" through a "hasColor" relationship somehow implies that sentences like "the blue lamp" should be correct in this domain and therefore accepted by the NLU grammar.

The generation of linguistic knowledge from ontologies has been proposed previously. Russ et al. [6] proposed a method for generating context–free grammar rules from JFACC ontologies. Their approach was based on including annotations all along the ontology indicating how to generate each rule. They implemented a program that was able to parse the ontology and produce the grammar rules.

A second precedent of linguistic generation from ontologies can be found in [7], where the author claimed that the concepts of an OWL ontology could be used to generate the lexicon of the NLU module.

In this paper a new rule–based solution for generating grammars from ontologies will be described. Section 2 motivates and gives an overview of the solution hereby proposed. Section 3 describes how the configuration files have to be built. Section 4 shows an introduction to the algorithm used. Section 5 includes real showcases of the tool at work. Section 6 is a summary of the conclusions and future work.

2 Solution overview

The solution proposed here is close to that of [6] in the sense that we also parse the ontology for the rule generation. Nonetheless, it differs in two ways:

Firstly, our approach argues that the ontology should remain as–is, without any specific linguistic annotation. Although it is obvious that the ontology itself is not descriptive enough to generate the grammar rules without additional information, we consider it preferable to place this information in a separate configuration file which describes how the grammar rules should be generated. This approach is also more convenient for a dialogue system (where the linguistic information in the ontology would be useless and cumbersome), and more suitable from a reusability point of view.

Secondly, OWL has been chosen instead of JFACC for two reasons:

1. The use of OWL is widely spread and seems to be the basis for the future semantic web. This implies that large ontologies are likely to be available in the future. Our approach will help create dialogue applications more easily by simply downloading specific domain ontologies.
2. OWL is based on RDF, and therefore uses Subject–Property–Object triplets. This static structure of OWL is of great help because the algorithm can focus on handling properties, letting the linguist define how to create rules which apply to all the elements in their Domain (or Range). It should be pointed out that this choice is not just a change in the ontology format: the whole parsing algorithm is based on RDFs predefined structure.

As previously mentioned, our approach focusses on grammar rules generation: no automatic lexicon hierarchy generation has been considered. To ensure coherence between the lexicon and the grammar, the list of potential non–terminal types is extracted from the list of all the entities within the ontology. The linguist decides which entities from this list shall remain in the final dialogue application.

It should be pointed out that this approach is meant only as a way of helping the linguist, and therefore it does not provide a ready–to–use grammar. By using this tool, the grammar will be easier to generate and more consistent with the domain knowledge, but, in any case, the resulting grammar must be checked and completed manually in a second step.

The current implementation provides grammar rules in a self–defined format [8] , which basically consists of a left–hand symbol followed by an arrow and a list of right–hand symbols. This notation can be easily translated to most standards, such as BNF.

3 Configuration files

As outlined above, the linguist must define a configuration file which will be used in conjunction with the ontology in order to generate the grammar rules. In this configuration file, the linguist has to identify the properties that may appear in the grammar and the way in which their domain and range will be included in the associated rules. In order to do it, an easy XML syntax has been defined (see DTD below).

Basically, the linguist can define the generation rules by means of nested *forEach* loops handling the properties (and subproperties) of the ontology, and using variables to identify the elements from its domain and range.

```
<!DOCTYPE rulesList [
  <!ELEMENT rulesList (forEach+)>
  <!ELEMENT forEach (forEach|rule+)>
  <!ELEMENT rule (left,right)>
  <!ELEMENT left (#PCDATA)>
  <!ELEMENT right (#PCDATA)>

  <!ATTLIST forEach property CDATA #IMPLIED>
```

```
  <!ATTLIST forEach subPropertyOf CDATA #IMPLIED>
  <!ATTLIST forEach domain CDATA #IMPLIED>
  <!ATTLIST forEach range CDATA #IMPLIED>

  <!ATTLIST rule lang (ES|EN|FR) #REQUIRED>
]>
```

In order to better understand this structure as well as the objective of the tool, a selection of showcases including those relevant in the ontology, the configuration file, and the resulting grammar rules are shown in the following sections.

4 Overview of the algorithm

This section describes in some detail the functions in the algorithm, which consists of three major steps:

1. Parse the OWL ontology. The goal of this step is to generate an internal representation of the relevant ontological elements. This representation will in turn be used to make queries over the ontology.
2. Parse the configuration file. The objective here is to generate the list of all applicable rules.
3. Generate the output rules. In this step, the script goes through the previous list of applicable rules, substituting the reference to classes and properties by the corresponding Input Form from the ontology.

The first two steps described have been implemented through a finite state machine (FSM) illustrated in figure 4.

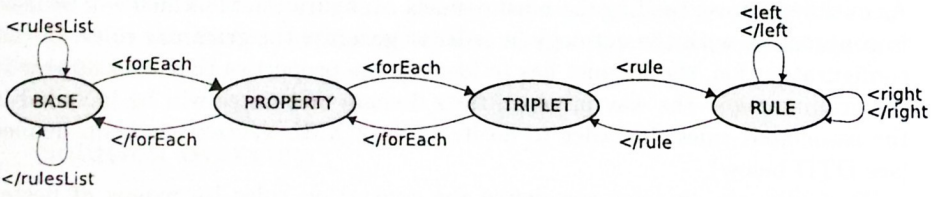


Fig. 1. FSM for the configuration file parser

For each state in the FSM, only one set of attributes can be parsed. These are mentioned in the previous DTD structure:

Base :
- No attributes are expected in this state.

Property :
- propertyRef: Indicates the word which refers to the property in the rule description.

- subPropertyOf: Indicates a super property. All the sub properties of this one will be handled by the algorithm.

Triplet :

- domainRef: Indicates the word which makes reference to the domain in the rule description.
- rangeRef: Indicates the word which refers to the range in the rule description.

Rule :

- lang: Indicates for which language the rule is valid.

5 Showcases

5.1 Sample Rules

The example below illustrates a common case in which the grammar rules will be generated. Our examples are taken from a smart–house domain in which the ontology describes both the hierarchy of devices in the house as well as the actions (or voice commands) which can be performed over those devices, such as "switch on the lamp in the kitchen". Thus, consider an ontology where a set of properties are grouped as subproperties of a general "hasDeviceCommand" property. These properties are shown graphically below:

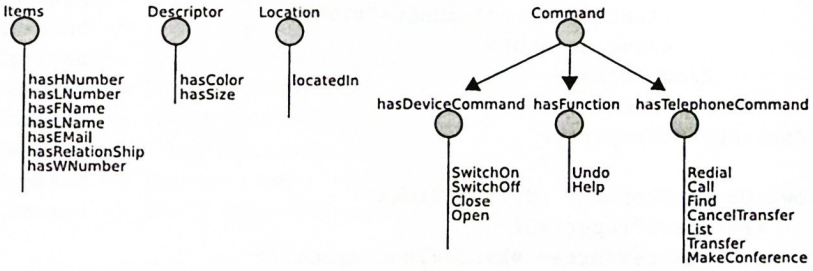


Fig. 2. Ontology Structure

In this showcase we are going to analyze the portion describing the device–related commands, whose XML equivalent is as follows:

```
<!-- hasDeviceCommand Subproperties -->

 <owl:ObjectProperty rdf:ID="SwitchOff">
  <rdfs:subPropertyOf
    rdf:resource="#hasDeviceCommand"/>
    <rdfs:domain rdf:resource="#System"/>
    <rdfs:range>
     <owl:Class>
```

```
      <owl:unionOf rdf:parseType="Collection">
         <owl:Class rdf:about="#Fan"/>
         <owl:Class rdf:about="#Heater"/>
         <owl:Class rdf:about="#Lamp"/>
         <owl:Class rdf:about="#Radio"/>
         <owl:Class rdf:about="#TV"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:range>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="SwitchOn">
    <rdfs:subPropertyOf
        rdf:resource="#hasDeviceCommand"/>
    <rdfs:domain rdf:resource="#System"/>
    <rdfs:range>
        <owl:Class>
          <owl:unionOf
            rdf:parseType="Collection">
            <owl:Class rdf:about="#Fan"/>
            <owl:Class rdf:about="#Heater"/>
            <owl:Class rdf:about="#Lamp"/>
            <owl:Class rdf:about="#Radio"/>
            <owl:Class rdf:about="#TV"/>
          </owl:unionOf>
        </owl:Class>
    </rdfs:range>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="Close">
    <rdfs:subPropertyOf
        rdf:resource="#hasDeviceCommand"/>
    <rdfs:domain rdf:resource="#System"/>
    <rdfs:range rdf:resource="#Blind"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="Open">
    <rdfs:subPropertyOf
        rdf:resource="#hasDeviceCommand"/>
    <rdfs:domain rdf:resource="#System"/>
    <rdfs:range rdf:resource="#Blind"/>
</owl:ObjectProperty>
```

In this particular case, the linguist has detected that all properties are actually actions, that is, they correspond to the "commands" to be performed by

the system over all the elements in the range, that is, all the devices within the ontology. This can be easily expressed by the following configuration file:

```
<rulesList>
  <forEach property="Z" subPropertyOf="hasDeviceCommand">
    <forEach domain="X" range="Y">
      <rule lang="ES">
        <left>Command</left>
        <right>Z Y</right>
      </rule>
    </forEach>
  </forEach>
</rulesList>
```

Now, once the application is run indicating the appropriate configuration file, the following results are obtained [1]

```
Command -> IForm_SwitchOff IForm_Fan
Command -> IForm_SwitchOff IForm_Heater
Command -> IForm_SwitchOff IForm_Lamp
Command -> IForm_SwitchOff IForm_DimmerLamp
Command -> IForm_SwitchOff IForm_Radio
Command -> IForm_SwitchOff IForm_TV
Command -> IForm_SwitchOn IForm_Fan
Command -> IForm_SwitchOn IForm_Heater
Command -> IForm_SwitchOn IForm_Lamp
Command -> IForm_SwitchOn IForm_DimmerLamp
Command -> IForm_SwitchOn IForm_Radio
Command -> IForm_SwitchOn IForm_TV
Command -> IForm_Close IForm_Blind
Command -> IForm_Open IForm_Blind
```

The grammar rules obtained are semantically driven, without purely linguistic items like "Noun" or "Preposition". This is typically the case of dialogue systems grammars.

It should be noticed that even with this toy ontology, sixteen grammar rules have been generated using just two nested *forEach* loops.

5.2 Capturing Multimodality

Now let us assume the same scenario (i.e. the same ontology) but including multimodal entries; namely voice and pen inputs. Following Oviatt's results [9], it may be expected that mixed input modalities (voice: *switch this on*, pen: `clicks`

[1] The prefix "IForm" stands for "Input Form", used to identify non-terminal symbols in our self-defined format [8] . This prefix has no bearing on the algorithm: any other token could be used.

on the lamp icon) may also include alternative constituent orders, that is, different from the voice only input. The NLU module may therefore receive inputs such as: "lamp switch on" (verb at the end). [2]

This new set of rules can be easily accounted for by adding just one rule to the configuration file:

```
<rulesList>
   <forEach property="P"
            subPropertyOf="hasDeviceCommand">
      <forEach domain="X" range="Y">
         <rule>
            <left>Command</left>
            <right>P Y</right>
         </rule>
         <rule>
            <left>Command</left>
            <right>Y P</right>
         </rule>
      </forEach>
   </forEach>
</rulesList>
```

The new output will be the same as before but including these new rules:

```
Command -> IForm_Fan IForm_SwitchOff
Command -> IForm_Heater IForm_SwitchOff
Command -> IForm_Lamp IForm_SwitchOff
Command -> IForm_DimmerLamp IForm_SwitchOff
Command -> IForm_Radio IForm_SwitchOff
Command -> IForm_TV IForm_SwitchOff
Command -> IForm_Fan IForm_SwitchOn
Command -> IForm_Heater IForm_SwitchOn
Command -> IForm_Lamp IForm_SwitchOn
Command -> IForm_DimmerLamp IForm_SwitchOn
Command -> IForm_Radio IForm_SwitchOn
Command -> IForm_TV IForm_SwitchOn
Command -> IForm_Blind IForm_Close
Command -> IForm_Blind IForm_Open
```

5.3 Capturing Multilinguality

Due to the structural differences among human languages, different rules must be generated for different languages.

[2] Note that linguistically speaking this order is also possible in English in topicalized or left-dislocated constructions such as "The lamp, switch it on".

For example, in order to indicate the location of a given device, "the kitchen light" is said in English, whereas in Spanish the constituent order changes: "la luz de la cocina" (literally, the light of the kitchen).

Once the target language has been chosen, specific language rules may be generated.

Consider the following fragment taken from the ontology previously shown, describing which elements can be affected by the property "locatedIn":

```
<owl:ObjectProperty rdf:ID="locatedIn">
    <rdfs:domain>
        <owl:Class>
            <owl:unionOf
                rdf:parseType="Collection">
                <owl:Class rdf:about="#Lamp"/>
                <owl:Class rdf:about="#Radio"/>
                <owl:Class rdf:about="#Heater"/>
             </owl:unionOf>
         </owl:Class>
     </rdfs:domain>
     <rdfs:range>
        <owl:Class>
         <owl:unionOf
          rdf:parseType="Collection">
            <owl:Class rdf:about="#Bedroom"/>
            <owl:Class rdf:about="#Kitchen"/>
            <owl:Class rdf:about="#Hall"/>
            <owl:Class rdf:about="#LivingRoom"/>
         </owl:unionOf>
        </owl:Class>
      </rdfs:range>
</owl:ObjectProperty>
```

The multilingual configuration file which captures the structural differences mentioned above would be the following:

```
<rulesList>
   <forEach property="P"
    subPropertyOf="Location">
      <forEach domain="X" range="Y">
         <rule lang="ES">
            <left>X</left>
            <right>X P Y</right>
         </rule>
        <rule lang="EN">
            <left>X</left>
            <right>Y X</right>
```

```
        </rule>
      </forEach>
    </forEach>
</rulesList>
```

Now, if only English grammar rules are to be generated, the application must be run with the option "-lang=EN", which obtains the following results:

```
IForm_Lamp -> IForm_Bedroom IForm_Lamp
IForm_Lamp -> IForm_Kitchen IForm_Lamp
IForm_Lamp -> IForm_Hall IForm_Lamp
IForm_Lamp -> IForm_LivingRoom IForm_Lamp
IForm_Radio -> IForm_Bedroom IForm_Radio
IForm_Radio -> IForm_Kitchen IForm_Radio
IForm_Radio -> IForm_Hall IForm_Radio
IForm_Radio -> IForm_LivingRoom IForm_Radio
IForm_Heater -> IForm_Bedroom IForm_Heater
IForm_Heater -> IForm_Kitchen IForm_Heater
IForm_Heater -> IForm_Hall IForm_Heater
IForm_Heater -> IForm_LivingRoom IForm_Heater
```

6 Conclusions and future work

In this paper a novel rule-based approach to automatic grammar generation has been described. The solution proposed is based on OWL ontologies and provides linguists with an easy way to take advantage of the information contained within ontologies. This information extraction process will also be easier for the linguist if the ontology has been designed keeping in mind that grammars will be generated from it.

The solution proposed has achieved the expected goals: the linguist can generate a good number of rules from a simple configuration file and, by having the rules directly generated from the ontologies, domain knowledge and linguistic knowledge coherence and completeness is ensured. In addition, a rapid prototyping of new grammars for the speech recognizer and the NLU module is obtained by the same mechanism.

Future research areas include the generation of unification-based grammar rules and dialogue rules, and an evaluation of the usefulness of the tool with larger OWL ontologies.

7 Acknowledgements

This work was done under the TALK research project, funded by EU FP6 [ref. 507802] and the "Multilingual Management of Spoken Dialogues" project, funded by the Spanish Ministry of Education under grant TIC2002-00526.

References

[1] Zaanen, M.V.: Abl: Alignment–based learning. In: Proceedings of the 18th International Conference on Computational Linguistics (COLING), Saarbrücken (2000)

[2] Williem Adriaans, P.: Language Learning from a Categorial Perspective. PhD thesis, Amsterdam University (1992)

[3] Ranta, A.: Grammatical framework. a type–theoretical grammar formalism. The Journal of Functional Programming **14** (2004) 145–189

[4] Milward, D., Beverige, M.: Ontology–based dialogue systems. In: IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems. (2003)

[5] Quesada, J.F., Amores, G.: Knowledge–based reference resolution for dialogue management in a home domain environment. In Johan Bos, M.E., Matheson, C., eds.: Proceedings of the sixth workshop on the semantics and pragmatics of dialogue (Edilog). (2002) 149–154

[6] Russ, T., Valente, A., MacGregor, R., Swartout, W.: Practical experiences in trading off ontology usability and reusability. In: Proceedings of the Knowledge Acquisition Workshop (KAW99), Banff, Alberta (1999)

[7] Estival, D., Nowak, C., Zschorn, A.: Towards ontology-based natural language processing. In: RDF/RDFS and OWL in Language Technology: 4th Workshop on NLP and XML, Barcelona, Spain, ACL (2004)

[8] Amores, G., Quesada, F.: Episteme. Procesamiento del Lenguaje Natural **21** (1997) 1–16

[9] Sharon Oviatt, S.L., DeAngeli, A., K., K.: Integration and synchronization of input modes during multimodal human–computer interaction. In: Proceedings of Conference on Human Factors in Computing Systems: CHI '97. (1997)

Clustering of English-Korean Translation Word Pairs Using Bi-grams

Hanmin Jung[1], Hee-Kwan Koo[2], Won-Kyung Sung[1] and Dong-In Park[1]

[1] NTIS Division, KISTI, Korea
jhm@kisti.re.kr
[2] Practical Information Science, UST, Korea

Abstract. This paper describes a clustering algorithm for Korean translation words automatically extracted from Korean newspapers. Since above 80% of English words appear with abbreviated forms in Korean newspapers, it is necessary to make the clusters of their Korean translation words to easily construct bi-lingual knowledge bases such as dictionaries and translation patterns. As a seed to acquire each translation cluster, we repeat to choose an adequate translation word from a remaining translation set using an extended bi-gram-based binary vector matching until the set becomes empty. We also deal with several phenomena such as transliterations and acronyms during the clustering. Experimental results showed that our algorithm is superior to Dice coefficient and Jaccard coefficient in both determining adequate translation words and clustering translations.

1 Introduction

As information technology develops in recent years, many terminologies are rapidly created and discarded. Newspapers are excellent resources to acquire new-coined terms and to inspect their life cycle [4]. About 90% of terms in Korean newspapers, in particular, are originated from foreign languages such as English and Chinese[1] [1]. Some of them are accompanied by original words in English for readers to easily grasp the meaning, for example, "세계무역기구 (WTO)." However, many English words (about 82% in our test set) appear with abbreviated forms, and translations differ like "아시아태평양경제협력기구," "아시아태평양경제협력체," "아태경제협력체," and "아태경제협력회의" for "APEC; Asia-Pacific Economic Cooperation." Such English abbreviated forms tend to cause word sense ambiguities, for example, "Internet Service Provider," "Information Strategic Planning," and "Image Signal Processor" for "ISP." Newspapers also usually use parentheses to represent a pair of translation pairs, but they are not limited to the pairs. Many extraction errors are caused by the free uses of parentheses such as "모델명 S3C2410 (CPU)"[2] and

[1] E.g., "아펙" is a Korean transliterated word for English "APEC," and "경제" is for Chinese "经济."
[2] "모델명 S3C2410" = "모델명 (Model No.)" + "S3C2410."

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 15-24

"경제한파 (IMF)."[3] Korean transliteration is another consideration for the design of a translation clustering model since it does not contain any translation meaning but imply pronunciation rules. These phenomena should be resolved to make translation clusters and to determine adequate translation words, which are crucial for the building of translation knowledge bases.

However, previous studies failed to notice the need for the clustering [4, 5]. They focused only on automatic transliteration and unabbreviated word translation. We think they might not collect and analyze the real status of a huge newspaper corpus. In this paper, we will introduce the subsequent methods to manage translations in a real newspaper corpus with the amount of about 30 million Korean words: transliteration clustering, translation clustering including acronyms, and adequate translation determination.

Automatic transliteration can be implemented by direct and pivot-based translation systems [6]. Previous studies tried to generate several possible candidate words based on pronunciation derived by dictionaries and statistical approaches such as Markov window and decision tree [4, 5, 6]. However, they considered only English unabbreviated words that generate many possible transliteration candidates. It is the reason why they introduced statistical methods to rank the candidates. Comparison of an English word with a Korean word is much easier than generating the best transliteration candidate for the given English word. In addition, the ratio of English abbreviated words in Korean newspaper corpus is above 80%, which indicates that complex pronunciations (e.g. "er" and "eo") appear less than unabbreviated words.

Example-based translation systems like [2] usually use linguistic information and statistical information. The number of element words in each language becomes a basic feature to acquire linguistic information. However, the number of element words in abbreviated forms cannot be directly calculated. Statistical information for corresponding probability is also meaningless because we extract bilingual words from translation patterns not bilingual corpus.

Important issues in our research scope are to make translation clusters and to determine an adequate translation word for each cluster from monolingual corpus. These are the points that our research scope differs from the alignment and the extraction of translation patterns from bilingual corpus [7, 8]. Unfortunately, there is no study of the issues for Korean newspaper corpus. Nobody tried to extract a set of Korean translations for an English word in a real newspaper. Ignoring English abbreviated forms that frequently appear in the corpus would be another reason to skip the issues.

We found that clustering method using similarity between surface forms is more efficient than using dictionaries and partial translation word matching since translation words appear with various forms and parentheses are widely used to clarify the meaning of the words. For example, Korean translations for "EC" are, at least, morphologically classified into three groups: "Electronic Commerce," "European Commission," and "Electrolytic Condensers." The whole process including an extended bi-gram-based binary vector matching to measure semantic distance between two translation words and to determine an adequate one for a cluster will be introduced in Section 2 and 3.

---

[3] "경제한파" = "경제 (Economic)" + "한파 (Cold wave)." "국제금융기구" is a right translation of "IMF."

2 System Overview

To generate translation clusters for an English word, we introduce four functions: *FindTransliterationCluster* (see Section 3.1), *DetermineAdequateTranslation* (see Section 3.2), *FindTranslationCluster* (see Section 3.3), and *FindAcronymCluster* (see Section 3.4). An adequate translation word is automatically obtained before generating a cluster for it.

```
TranslationClustering (T) {
        C1 = FindTransliterationCluster (T);
        T = T – C1;
        i = 2;
        Repeat while T is not NULL {
                Ci.adequate_translation = DetermineAdequateTranslation (T);
                Ci = FindTranslationCluster (Ci.adequate_translation, T);
                T = T – Ci;
                Ci = Ci + FindAcronymCluster (Ci.adequate_translation, T);
                T = T – Ci;
                Increase i;
        }
        Return C;
}
```

Fig. 1. Translation clustering process including transliterations and acronyms (Both T and C_i are the sets of translations, and C_i.adequate_translation is a translation word.)

T is the Korean translation set of an English word. It includes one or more translation clusters that will be found as the above process goes ahead. A translation cluster consists of Korean translation words with the same meaning. *TranslationClustering* finds these clusters C_1, C_2, and so on (see Section 3). *FindTransliterationCluster* generates a translation cluster whose components are transliterations, for example, "시스템온칩" and "플랫폼[4]." Since they have no meaning in Korean, we separate them as a distinct cluster (C_1) from translation set T (see Section 3.3). The loop to find translation clusters continues until the translation set T becomes NULL. Whenever iteration ends, we find a translation cluster including an adequate translation word in it. *DetermineAdequateTranslation* gives us the adequate translation word, that is, the word with the most shared bi-grams in translation set T (see Section 3.2). *FindTranslationCluster* generates a translation cluster in the manner of matching the adequate (C_i.adequate_translation) with the translation words in set T. In the case that a Korean word shares one or more bi-grams with the adequate, we consider the two translations are in the same translation cluster. *FindAcronymCluster* discovers acronyms for the adequate (C_i.adequate_translation). Finally, we acquire a set of translation clusters ($C = \{C_1, C_2 \ldots\}$).

---

[4] "시스템온칩" = "시스템 (System)" + "온 (On)" + "칩 (Chip)"
"플랫폼" = "플랫폼 (Platform)"

Table 1. An example to acquire translation clusters for English word "KAIST; Korea Advanced Institute of Science and Technology" (Underlined are newly added terms.)

Initial State	
T	{한국과학기술원, 한국과학기술기술원, 연기한국과학기술원, 카이스트, 과기원, 나노종합팩, 석사·박사[5]}
After FindTransliterationCluster	
C_1	{카이스트}
1st Iteration	
After DetermineAdequateTranslation	
C_2.adequate_translation	한국과학기술원[6]
After FindTranslationCluster	
C_2	{한국과학기술원, 한국과학기술기술원, 연기한국과학기술원}
After FindAcronymCluster	
C_2	{한국과학기술원, 한국과학기술기술원, 연기한국과학기술원, 과기원[7]}
2nd Iteration	
After DetermineAdequateTranslation	
C_3.adequate_translation	나노종합팩
After FindTranslationCluster	
C_3	{나노종합팩}
3rd Iteration	
After DetermineAdequateTranslation	
C_4.adequate_translation	석사·박사
After FindTranslationCluster	
C_4	{석사·박사}
Translation Clusters	
C	{C_1, C_2, C_3, C_4}

3 Translation-Clustering

3.1 Finding a Transliteration Cluster

In Korean, there are two ways to make translation words; one is transliteration (e.g. "이미지시그널프로세싱[8]") and the other is liberal translation using Chinese characters (e.g. "영상신호처리[9]"). Transliterations should not be assigned into other trans-

---

5 The translation set T includes an extraction error "연기한국과학기술원" and a typing error "한국과학기술기술원." "나노종합팩" and "석사·박사" are a little irrelevant terms with "KAIST."

6 한국과학기술원 (韓國科學技術院). It is the best translation in Korean.

7 과기원 (科技院). It is an acronym of "과학기술원."

8 "Image Signal Processing" ↔ "이미지시그널프로세싱" = "이미지 (Image)" + "시그널 (Signal)" + "프로세싱 (Processing)"

9 "Image Signal Processing" ↔ "영상신호처리" = "영상[映像] (Image)" + "신호[信號] (Signal)" + "처리[處理] (Processing)"

lation clusters since they have no meaning in Korean. Thus, we generate a separate translation cluster for these by applying *FindTransliterationCluster*.

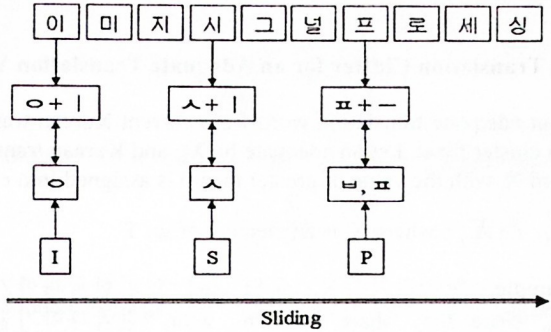


Fig. 2. An example to compare an English word "ISP" with a Korean transliteration candidate "이미지시그널프로세싱"

Unlike [4] and [5] which tried to automatically generate the best transliteration for an English word, it is easier to determine a translation word whether it is a transliteration or not. We make a set of simple mapping rules that each contains a possible Korean alphabet[10] list corresponding to an English character, for example, {'ㅂ,' 'ㅍ'} for 'p.' We convert each character of an English word into a sequence of Korean alphabet lists. Figure 2 shows an example of this mapping. It is very similar to the acronym-finding process of Section 3.3.

3.2 Choosing an Adequate Translation Word

Let the adequate translation word of a translation cluster C_i be C_i.adequate_translation whose adequate-value (AV) is the maximum in translation set T. AV_k is for a translation word k in set T, and is defined by the subsequent equation (1). The number of translation words in set T is n. AV_k increases as k shares bi-grams with the other words more and more. We prefer a shorter word to a longer word when tie occurs.

$$AV_k = \frac{\sum_{j=1}^{n} |X_k \cap X_j|}{|X_k|} \quad \text{where } k \neq j \tag{1}$$

Let us show an example to determine an adequate one for a translation set {"한국과학기술원," "한국과학기술기술원," "연기한국과학기술원," "과기원"}. "한국과학기술원" shares 6 bi-grams with "한국과학기술기술원,"[11] and 6 with "연기한국과학기술원."[12] Since its bi-gram length is 6, AV becomes 2. The ade-

[10] A Korean alphabet is a phonetic unit that can be a consonant or a vowel.

[11] The shared bi-grams are "한국," "국과," "과학," "학기," "기술," and "술원."

[12] The shared bi-grams are the same as the above.

quate-value of "한국과학기술원" is the maximum among the others, thus it is chosen as the adequate translation word from the example set. A translation cluster then is generated from the set by matching with the adequate word as follows.

3.3 Finding a Translation Cluster for an Adequate Translation Word

After obtaining an adequate translation word from current Korean translation set, we find a translation cluster for it. Let an adequate be X_{C_i} and Korean translation set be T. A translation word X_j with the value of greater than 0 is assigned into cluster C_i.

$$| X_{C_i} \cap X_j | \text{ where } X_j \text{ is an element of set T} \quad (2)$$

In the above example, "한국과학기술기술원" and "연기한국과학기술원" become members of C_i, since they share bi-grams with "한국과학기술원" which is C_i.adequate_translation.

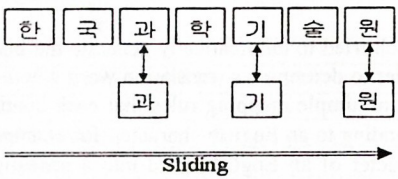


Fig. 3. An example to find an acronym ("과기원") using a Korean unabbreviated word ("한국과학기술 원") which is an adequate translation word.

3.4 Finding an Acronym Cluster

Some Korean unabbreviated words, in particular, organization names, written in Chinese characters[13] have acronyms such as "한국과학기술원 (韓國科學技術院) → 과기원 (科技院)" and "정보통신부 (情報通信部) → 정통부 (情通部)." As a Korean unabbreviated word and its acronym have the same character sequences, we can easily match the two words in the manner of left-to-right scanning. Matched acronyms then are assigned into the previously acquired translation cluster (see C_2 in Table 1).

4 Experimental Results

From Korean IT newspaper corpus,[14] we extracted Korean-English pairs combined with parentheses such as "북미자유무역협정 (NAFTA)" and "나프타 (NAFTA)."

[13] Most of the Korean words are originated from Chinese.

[14] Electronic Times (http://www.etnews.co.kr/)

Total 1,806 Korean translation sets were acquired after re-arranging on English words, and 200[15] of them were used to measure the subsequent performance.

We choose Dice coefficient as a criterion to compare with our method, and modify it like

$$AV_k = \sum_{j=1}^{n} \frac{2|X_k \cap X_j|}{|X_k| + |X_j|}$$

where $k \neq j$ (*see Section 3.2 to refer the notations*). Jaccard coefficient

$$AV_k = \sum_{j=1}^{n} \frac{|X_k \cap X_j|}{|X_k \cup X_j|}$$

where $k \neq j$, is another criterion. As these two algorithms are bi-gram approaches to easily calculate the similarity between strings, they were chosen. Korean word X_k would be selected when it has the highest AV value. According to our clustering algorithms, different adequate translation words have different clusters.

To measure the performance, we manually attached cluster tags to the above-mentioned 200 translation sets. A translation set consists of one or more semantically separate clusters without consideration of surface forms. The subsequent shows an example of the tagging results for our answer set (C1, C2, and C3 are cluster tags, and A is answer adequate translation word.). Adequate words can be multiple in a cluster.

[ATM] 현금자동입출금기/C1/A 현금입출금기/C1/A
초대형현금자동입출금기/C1 자동화기기/C1
비동기전송모드/C2/A 비동기전송방식/C2/A 장비-비동기전송모드/C2
초고속정보통신망/C3/A 초고속국가망/C3
초고속교환기/C3 초고속국가정보통신/C3[16]

Table 2 shows the comparison between our methods and the Dice/Jaccard Coefficients. Precision for choosing adequate translation word is the ratio of the number of correct words to the number of chosen adequate words. The system automatically checks whether adequate words and clusters are correct or not by comparing with the answer set. In the case that one or more translation words in an acquired cluster have different cluster tags with the other words in the cluster, we consider the cluster is wrong. Recall for clustering translations is the ratio of the number of correct clusters to the number of answer clusters. A cluster is recognized as correct one if and only if all the translation words of it are exactly matched with those of a cluster in the answer set.

---

[15] We selected translation sets with more than five translation words. The number of total words is 2,253 and the average number of translation words for a translation set is 11.265.

[16] The words with C3 are not translation words of "ATM." However, we always attached answer tags because our research topic does not concern about the determination of whether a word really is translation word or not.

Table 2. Comparison among our method, Dice coefficient, and Jaccard coefficient[17]

	Our Method	Dice Coefficient	Jaccard Coefficient
The Number of Translation Clusters	617	623	623
Average Size of Translation Clusters	3.651	3.617	3.617
Recall for Choosing Adequate Translation Word	82.496% (575/697)	75.036% (523/697)	75.036% (523/697)
Precision for Choosing Adequate Translation Word	93.193% (575/617)	83.949% (523/623)	83.949% (523/623)
F-measure for Choosing Adequate Translation Word	87.519%	79.243%	79.243%
Recall for Clustering Translations[18]	64.275% (448/697)	61.549% (429/697)	61.549% (429/697)
Precision for Clustering Translations	72.609% (448/617)	68.104% (429/623)	68.104% (429/623)
F-measure for Clustering Translations	68.188%	64.661%	64.661%

The reason why our method shows higher performance than the other two is that we discriminatively apply length information to eliminate superfluous words attached adequate translation word due to automatic extraction from corpus, for example, "로열티한국전자통신연구원 (ETRI)" and "셀러론중앙처리장치 (CPU)"[19]. These redundancies can be easily eliminated since they appear rarely.

We found three factors that decrease the performance: word sense ambiguities of English abbreviated words, synonyms without sharing bi-gram, and fake translation pairs with parentheses. (1) "WTO" has two meanings: "World Trade Organization" and "World Tourism Organization." Their representative translation words are "세계무역기구" and "세계관광기구." They share "세계" and "기구," thus the two translations are recognized as the same members of a cluster by our algorithms. Some English abbreviated words including widely used components such as "system" and "technology" tend to have many word sense ambiguities. Using stop words for them would be helpful to reduce the ambiguities. (2) "IPO" as "Initial Public Offering" has several translation words with the same meaning such as "기업공개" and "주식공모," even though they do not share any bi-gram. Introducing morphological analysis and synonym set (e.g. "공개=공모" and "미=미국"[20]) would be helpful to

[17] It is interesting that the two coefficients show the same performance even though adequate values are different.

[18] Each translation set has one or more translation clusters, and even garbage clusters including extraction errors, because of translation ambiguity. This is the reason why the number of correct translation clusters is 697 not 200.

[19] "로열티" is for "Royalty" and "셀러론" is for "Celeron."

[20] "미 (美)" is a Korean abbreviated form of "미국 (美國)." Both words are represented for "USA."

enhance the clustering performance. (3) As previously mentioned, newspapers widely use parentheses to expatiate translation words. The pairs can accidentally share bi-grams with other translation pairs, for example, "삼성전자 (ETRI)" and "한국전자통신연구원 (ETRI)."[21] It will be a way to reduce wrongly extracted translation pairs by referring English unabbreviated words corresponding to English abbreviated forms.

5 Conclusions

We introduced a practical translation-clustering algorithm for translation pairs automatically extracted from newspaper corpus by using an extended bi-gram-based binary vector matching. To increase clustering coverage, our research scope included transliterations and acronyms. It has an important meaning in that previous studies could not consider a great portion of abbreviated forms appeared in a real newspaper corpus. Knowledge builders can easily confirm the clustering results because the system shows both adequate translation words and translation clusters. We now consider introducing cluster verification by Web search and histogram-based adequate translation determination as future works.

References

1. Choi. K. and Chae, Y.: Terminology in KOREA: KORTERM. Proceedings of LREC-2000.
2. Izuha, T.: Machine Translation Using Bilingual Term Entries Extracted from Parallel Texts. Journal of Systems and Computers. Vol.36 No.8 (2005)
3. Jung, H., Koo, H., Lee, B., and Sung, W.: Toward Managing the Life Cycle of Terms Using Term Dominance Trend. Proceedings of the Pacific Association for Computational Linguistics (2005)
4. Jung, S., Hong, H., and Baek, E.: An English to Korean Transliteration Model of Extended Markov Window. Proceedings of the 18th conference on Computational linguistics (2000)
5. Lee, J.: An English-Korean Transliteration and Retransliteration Model for Cross-lingual Information Retrieval. Ph.D. Thesis. Korea Advanced Institute of Science and Technology (1999)
6. Oh, J. And Choi, K.: An English-Korean Transliteration Model Using Pronunciation and Contextual Rules. Proceedings of the International Conference on Computational Linguistics (2002)
7. Ohara, M., Matsubara, S., and Inagaki, Y.: Automatic Extraction of Translation Patterns from Bilingual Legal Corpus. Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering (2003)
8. Tufis, D., Barbu, A., and Ion, R.: Extracting Multilingual Lexicons from Parallel Corpora. Journal of Computers and Humanities. Vol.38 No.2 (2004)

[21] "삼성전자" is for "Samsung Electronics Inc." and "한국전자통신연구원" is for "Electronics and Telecommunications Research Institute."

Morphology and Syntax

Modified Makagonov's Method for Testing Word Similarity and its Application to Constructing Word Frequency Lists

Xavier Blanco,[1] Mikhail Alexandrov,[1,2] and Alexander Gelbukh [2]

[1] Department of French and Romance Philology, Autonomous University of Barcelona
dyner1950@mail.ru, Xavier.Blanco@uab.es

[2] Center for Computing Research, National Polytechnic Institute (IPN). Mexico
dyner@cic.ipn.mx; www.Gelbukh.com

Abstract. By (morphologically) similar wordforms we understand wordforms (strings) that have the same base meaning (roughly, the same root), such as *sadly* and *sadden*. The task of deciding whether two given strings are similar (in this sense) has numerous applications in text processing, e.g., in information retrieval, for which usually stemming is employed as an intermediate step. Makagonov has suggested a weakly supervised approach for testing word similarity, based on empirical formulae comparing the number of equal and different letters in the two strings. This method gives good results on English, Russian, and a number of Romance languages. However, his approach does not deal well with slight morphological alterations in the stem, such as Spanish *pensar* vs. *pienso*. We propose a simple modification of the method using n-grams instead of letters. We also consider four algorithms for compiling a word frequency list relying on these formulae. Examples from Spanish and English are presented.

1 Introduction

Given a large text or text corpus and a pair of wordforms (strings), we consider the task of guessing whether these two words have the same root and thus the same base meaning. We call this (morphological) *word similarity*: two words are *similar* if they have the same root. This relation permits grouping together the words having the same root, e.g., *sad*, *sadly*, *sadness*, *sadden*, *saddened*, etc. This task has numerous applications, such as constructing word frequency lists. Our motivation is to improve information retrieval and similar practical applications. Consequently, our goal is to provide a reasonably accurate statistical-based algorithm (tolerating certain error rate) and not a precise linguistic analysis.

For grouping together the words with the same root, two morphology-based methods are usually used: lemmatization and stemming. Lemmatization reduces words to the base form: *having* → *have*; stemming truncates words to their stems: *having* → *hav-* (often lemmatization task is also referred to as stemming).

Stemming or lemmatization can be used for testing the (morphological) similarity between two words: both words are first reduced to lemmas or stems; if the resulting strings are equal then the two given words are declared similar. This gives a symmet-

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 27-36

ric, reflexive, and transitive relation that can be perfectly used for grouping together words with the same base meaning.

The stemming or lemmatizing algorithms using morphological rules and a large morphological dictionary provide practically 100% accuracy on known words and good accuracy on the words absent in the dictionary [4, 5]. The algorithms relying only on lists of suffixes and suffix removal rules, but no dictionaries, in spite of being much simpler, provide relatively high accuracy, often greater than 90%. The most popular algorithm is Porter stemmer [9] (actually, a lemmatizer). Both methods are strongly language-dependent. This becomes a problem in large-scale analysis of multilingual, multi-thematic document collections.

Makagonov *et al.* [7] suggested another approach to testing word similarity. It does not rely on a (language-dependent) intermediate step of reducing the two given words to a stem or lemma. Instead, it uses empirical formulae to compare the given strings directly. These formulae use the number of the coincident initial letters and non-coincident final letters in the two words. Such formulae are constructed by Ivakhnenko's [6] inductive method of model self-organization. In our paper [1] we have given more details on this method, investigated the sensibility of the formulae to different languages, and analyzed the typical errors of the method. The main advantage of this knowledge-poor approach is its simplicity and flexibility. It does not rely on any manually compiled morphological rules, dictionaries, suffix lists, or rule systems. To adapt the method to a new language or a new subject domain, only the parameters of the formula are to be automatically adjusted.

However, this method is sensitive to small differences in initial parts of similar words: an additional letter in one of these words may prevent the method from detecting their similarity. This affects mostly Romance languages with their irregular verbs: e.g., Spanish: *pienso* '(I) think' – *pensaba* 'thought', *entiendo* '(I) understand' – *entender* '(to) understand'. Obviously, this increases the rate of false negatives (failing to give a positive answer on a pair of words that are in fact similar). To avoid this, we propose here to use n-grams (3-grams) instead of letters in the original formula. N-grams have been already used for comparing words [10]. However, the work [10] concerns only evaluation of word importance but not their similarity. Nevertheless, it stimulated the research presented in this paper.

Another drawback of the approach from [7] is that the obtained relation is not transitive: if the formula reports the words a and b to be similar, as well as b and c, it does not necessarily report a and c to be similar. Strictly speaking, this prevents from using such a relation to group words together. However, [7] suggested a heuristic algorithm relying on these formulae for constructing word frequency lists (actually, for grouping words; the algorithm gives different results than a simple transitive closure).

We found, though, that this algorithm gives higher level of false negatives in comparison with the formulae themselves. We propose other algorithms, taking into account mentioned non-transitivity of our heuristic relation.

The paper is organized as follows. In Section 2, we explain the original algorithm and its empirical formulae. In Section 3, we present our modifications to the formula. We formally define the notion of n-gram as used by our algorithm and operations on such n-grams and then present the new formula and its training process. In Section 4, we introduce four algorithms for constructing word frequency lists and present experimental results. Section 5 concludes the paper and mentions some future work.

2 Testing Word Similarity Using Letters

2.1 Empirical Formulae

The empirical formulae from [7] test a hypothesis about word similarity. The proposed approach is applied only for suffixal inflective languages, i.e., the languages where the word base (morphologically invariant part) is located at the beginning of the word—which is generally true for the majority of European languages.

The formula relies on the following characteristics of the pair of words:

y: the length of the common initial substring (*y* standing for *yes*),
n: the total number of final letters differing in the two words (*n* for *no*),
s: the total number of letters in the two words (*s* for *sum*),

so that $2y + n = s$. For example, for the words *sadly* and *sadness*, the maximal common initial substring is *sad-*, thus the differing final parts are *-ly* and *-ness*, so that $n = 6$, $y = 3$, and $s = 12$.

The authors of [7] considered the following class of models for making decisions about word similarity: two words are similar if and only if

$$\frac{n}{s} \le F(y), \qquad F(y) = a + b_1 y + b_2 y^2 + \ldots + b_k y^k, \tag{1}$$

where $F(y)$ is the model function; a and b_i are constants (parameters of the formula). Such a function is general enough because any continuous function can be represented as a convergent polynomial series. The parameter k represents the model complexity.

To find the best model function, i.e., the model of optimum complexity, [7] used the inductive method of model self-organization. This method compares step-by-step the models of increasing complexity and stops this process when the optimum of an external criterion of model quality is reached [6]. This method relies on a set of examples, which are used to calculate the model parameters and evaluate the model quality; thus, the method is weakly supervised since the requited set of examples is very small. These examples are prepared by the user of the program for a specific language or genere.

With this method, the formulae for testing word similarity shown in Table 1 were constructed for different languages [1].

Table 1. Formulae for different languages

French	Italian	Portuguese	Spanish
$n/s \le 0.48 - 0.024\,y$	$n/s \le 0.57 - 0.035\,y$	$n/s \le 0.53 - 0.029\,y$	$n/s \le 0.55 - 0.029\,y$

Basing on all examples prepared for four languages, the authors determined the generalized formula:

$$n / s \le 0.53 - 0.029\,y \tag{2}$$

This formula can be considered an initial approximation for further tuning on other romance languages.

2.2 Discussion

Since the empirical formula is based on statistical regularities of a language, it leads to the errors of the two kinds (false positive and false negative; this can be rephrased in terms of precision and recall—see Sections 3.4 and 4.2). Varying the threshold function F between –1 and +1 we can control the balance between precision and recall. Our goal is to find the function that gives their acceptable combination; for example, the formula (2) gives rather acceptable results. Note that the formula's parameters depend not only on the language but also on a specific genre or domain. E.g., the formula $n/s \le 0.55 - 0.029\,y$ is optimal for general lexis in Spanish; however, for the texts on mortgage and crediting the best parameters proved to be $n/s \le 0.53 - 0.026\,y$.

Certain questions arise as to the obtained model function

$$\frac{n}{s} \le a + by \tag{3}$$

where $a > 0$, $b < 0$. This is a linear formula with two degrees of liberty where the threshold (the right-hand part) is lower for longer words. What does it mean from linguistics point of view?

(1) Our formula has in fact two degrees of liberty with respect to the parameters n, y, and s, since $s = n + 2y$. One can also consider a model in the form $n/s \le F(y/s)$, which has only one degree of liberty since $y / s = (1 - n/s) / 2$. It allows taking into account separately the statistics of both the initial and the final part of a given word.

(2) The linear approximation of the original formula $F(y) \sim a + by$ indicates high complexity of the real model we want to evaluate. Our model can reflect only the tendency (b is the first derivative of the model function) but not the shape of the function.

(3) All obtained formulas give lower threshold for longer words with the same relative number of non-coincident letters. This discrimination of long words reflects the following statistical property of a language: the length of the final part of similar words on average is similar for both long and short words. This property was noted in [8] and makes sense linguistically: the length of word endings (non-root morphemes) is the same for long and short words.

3 Testing Word Similarity Using *n*-grams

3.1 Limitation of the Original Approach

Our approach for testing word similarity is not applicable to words with irregular forms. Indeed, no simple formula can detect any similarity between irregular verbs such as *buy* and *bought*, because these words have only one common letter in the initial part of the string. Similar examples are there in Romance languages, e.g., Spanish *saber* 'know' vs. *supo* 'knew'. Obviously, this limitation leads to false negatives.

The other difficulty is related with the rigid comparison used in the formula. Namely, 'common part' means that both words have exactly equal initial substrings; slight changes in the common part dramatically reduce the size of the initial common

substring recognized by the formula, so word similarity is not detected, e.g.: Spanish *pienso* vs. *pensar*, *parezco* and *parecer*.

In the paper, we address the latter problem. We assume that the common initial part of both words may have small differences. In the paper, we limit these differences to one letter.

3.2 3-grams and Operations with them

Speaking about possible differences in common part of two words, we face a task of determination of the boundary of this common part. If we deal with one-letter "defects", then this task can be solved with 3-grams.

Definition 1. N-gram of letters is a substring of N letters. A word of n has of $m = n - N + 1$ N-grams. Example: The 3-grams for the word *revolution* are {rev, evo, vol, olu, lut, uti, tio, ion}.

Definition 2. N-gram associated with i-th position of a given word w is N-gram starting at i-th position. Truncated N-grams are added by attaching new "undefined" symbols to the words. All undefined symbols are supposed to be different, even though we denote them with the same letter X; i.e., $X \neq X$. A word of n-letters has exactly n different associated N-grams. Example: The 3-grams associated with the first and last letters of the word *revolution* are {rev, nXX}. Here X are the "undefined" symbols.

Definition 3. Full set of N-grams of letters for a given word w is all associated N-grams of this word. Example: The full set of 3-grams for the word *bill* are {bil, ill, llX, lXX}.

We will work with 3-grams, though similar definitions can be introduced for any N-grams. All comparisons are lexicographic. The definitions below can be rephrased in terms of Levenshtein distance, though we find them easier to understand in the form given here.

Definition 4. 3-grams ***A*** and ***B*** are equal with the level 1 if they are equal as strings. Example: 3-grams *pas* and *pas* are equal, while *pas* and *sap* are not.

Definition 5. 3-grams ***A*** and ***B*** are equal with the level 2/3 if two letters of one of them are found in the other one in the same order. Examples: 3-grams *pas* and *asp* as well as *ars* and *aps* are equal with the level 2/3, while *sra* and *aps* are not.

Definition 6. 3-grams ***A*** and ***B*** are equal with the level 1/3 if a letter of one of them is found in the other one. Examples: 3-grams *sra* and *ars* (*sra* and *ars*, *sra* and *ars*) are equal with the level 1/3, while *pas* and *wre* are not. Note that in the former case, both 3-grams contain the same letters but their order does not allow for equality with levels 2/3 or 1.

3-grams with undefined symbols X are compared taking into account that all such symbols are different: $X \neq X$. Examples:

- 3-grams *kl*X and X*kl* are equal with the level 2/3.
- 3-grams *kl*X and *k*XX are equal with the level 1/3.
- 3-grams XXX and XXX are not equal.

Definition 7. Two strings are equal by 3-grams with the level Q if they have equal 3-grams for all positions with the level Q. Obviously, if $Q = 1$ then these two strings are just equal as strings.

Examples: Strings *assenr* and *reasse* are equal by 3-grams with the level 1/3 and *assenr* and *yssien* with the level 2/3.

3.3 Constructing of Empirical Formula for 3-grams

We use the model described in Section 2.1, but instead of letters, our model operates on 3-grams. I.e., we declare two words similar if and only if

$$\frac{n}{s} \le F(y) \tag{4}$$

where y is the total number of 3-grams of the common *initial* substring, n is the total number of 3-grams in *final* substrings of the two words, s ia the total number of 3-grams in the two words, and F is the model function, so that $2y + n = s$. Since we use here all associated 3-grams (see Definition 3), the total number of n-grams is equal to the total number of letters in the two words.

Common substring is defined as the maximum common part of the two initial substrings, which have the same first letter and which are equal by 3-grams with the level no less than 2/3 (Definition 7). According to Definitions 4 and 5, this allows having one incompatible letter in the common part.

Examples:

a) For *sadly* and *sadness* (Section 2.1), 3-grams are: *sadly* = {*sad, adl, dly, lyX, yXX*}, *sadness* = {*sad, adn, dne, nes, ess, ssX, sXX* }; $n = 8, s = 12, y = 2$.
b) For Spanish *comiendo* and *comer*, 3-grams are: *comiendo* = {*com, omi, mie, ien, end, ndo, doX, oXX*}, *comer* = {*com, ome, mer, erX, rXX*}; $n = 7, s = 13, y = 3$.

In this paper we will not try to find the optimum shape of the model function F by the inductive method of model self-organization. Instead, extrapolating the results of experiments with traditional model, we will assume that the model to be constructed will be the linear. Thus, we will look for the optimum parameters for the formula (3).

3.4 Training Procedure for Spanish

The formula to be constructed has two unknown parameters, a and b. To find them we should prepare the set of examples and use the least squares method. It is well known from the theory of experiments that the number of examples must be at least 3 times greater then the number of parameters to be evaluated, which provides the relative error of 10%. Thus for our case we will need at least 6 pairs of similar words.

The examples we used for model construction are pairs of similar words with: (a) short and long initial common part, (b) short and long final parts, and (c) "defects" at the beginning and at the end of words; see Table 2.

The solution of this system gives the criterion:

$$n / s \le 0.63 - 0.036\, y \tag{5}$$

Table 2. Examples for training formula.

Examples		Parameters	Equations
Circo	*Circense*	$n = 7,\ s = 13,\ y = 3$	$7/13 = a + 3b$
Creado	*Creacion*	$n = 8,\ s = 14,\ y = 3$	$8/14 = a + 3b$
Sentimentales	*Sentimentalismo*	$n = 8,\ s = 28,\ y = 10$	$8/28 = a + 10b$
Necesario	*Necesariamente*	$n = 9,\ s = 23,\ y = 7$	$9/23 = a + 7b$
Pensar	*Pienso*	$n = 6,\ s = 12,\ y = 3$	$6/12 = a + 3b$
Entender	*Entiendo*	$n = 6,\ s = 16,\ y = 5$	$6/16 = a + 5b$

This formula should be considered only as a first approximation and may be later tuned on the texts from a given domain, since our examples selected for training formula only reflect some general regularities of a language.

We compared the work of traditional and modified algorithm on document collection on economic problems; see Table 3. The total number of words considered was 320 (numbers and words with less than 4 letters were excluded). The original algorithm used the formula (2); the algorithm based on 3-grams used the formula (5). The algorithm compared the pairs of words adjacent in the alphabetically ordered list, i.e., 319 comparisons were made. By false positive (negative) rate P_p (P_n) we understand the number of pairs incorrectly reported by the program as similar (not similar), divided by the number of really similar pairs in the corpus. Viewing the task as retrieval of similar pairs among all considered pairs, we can also represent the quality of the algorithm via precision P and recall R; obviously, $R = 1 - P_n$; $P = R / (R + P_p)$.

Table 3. Comparison of both methods

		Traditional algorithm	Modified algorithm
False positive	P_p	4.9%	5.4%
False negative	P_n	12.9%	10.6%
Total errors	P_{err}	17.8%	16.0%
Precision	P	94.7%	94.3%
Recall	R	87.1%	89.4%
F-measure		90.7%	91.8%

3.5 Discussion

We did not consider the words containing less than 4 letters, since these are mostly prepositions, conjunctives, and pronouns.

We neither considered the other n-grams, for example, 2-grams, 4-grams, etc. The only reason was that we wanted to implement the simplest principle of voting while detecting one-letter "defects". 3-grams were the minimum n-gram that allows doing it. However, in the future it is necessary to check N-grams with other N.

The formula we used reflected the statistical regularities of a language, with the error rate given in Table 3. By using n-grams we tried to reveal similar words having a "defect" in their common part, which led to decreased rate of false negatives. Although false positives rate slightly increased, the experiments showed that the overall error rate decreased.

4 Constructing Word Frequency Lists

4.1 Main Algorithms

The first algorithm oriented on application of empirical formula was very simple and consisted of the following steps [7]. Initially, the text collection is considered as a bag of words. With every word in this sequence, a counter is associated and initially set to 1. The algorithm proceeds as follows:

1. All words are ordered alphabetically; literally equal words are joined together, and their counts are summed up (e.g., 3 occurrences of the string *ask* with counters 2, 3, and 1 are replaced by one occurrence with the counter 6).
2. The similarity for each pair of adjacent words is tested according the criterion described above; namely, the 1-st word is compared with the 2-nd one, 3-rd with the 4-th, etc. If a pair of words is similar then these two words are replaced with one new "word"—their common initial part, with the counter set to the sum of the counters of the two original words. If the list has an odd number of words, then the last word is compared with the immediately preceding one (or with the result of substitution of the last word pair).
3. Step 2 is repeated until no changes are made at Step 2.

This multi-pass algorithm worked quickly but it proved to have a defect: it often omitted similar words in adjacent pairs. Therefore, we had to consider the algorithm in more detail. Note that the word similarity relation implemented in the present paper is not transitive, i.e., that two words are similar to a third one does not mean that the first two ones are similar. Thus, our algorithms for constructing word frequency lists tare sensitive to the order of comparison. We consider here two algorithms, which are applied to lists of alphabetically ordered words:

1. Algorithm where joining of adjacent similar words is used;
2. Algorithm where both adjacent and not adjacent words are considered.

Both algorithms start with Step 1 of the algorithm described above.

Algorithm 1

It is one-pass algorithm consisting of the following steps:

1. Starting from the first word from the list, the algorithm searches for a pair of similar words, considering the words from the next one just after it.
   - If such a pair is not found, then the algorithm stops.
   - Otherwise, these two words are substituted by a new one—their initial common part. Its counter is the sum of the counters of the two joined words.
2. The procedure is repeated with the next position.

Algorithm 2

It is a multi-pass algorithm, which checks for similarity of each word to each another word. It consists of the following steps:

1. Starting from the first word in the list, the algorithm searches for the first similar word among all words with the same initial letter (not necessary adjacent).
   - If such a word is not found, the process is repeated from the second word.
   - In case of success, the first word is substituted by the new one—their common initial part. Its counter is the sum of the counters of the two joined words. The second word from the pair is eliminated from the list. From the position of the eliminated word, the algorithm searches for a word similar to the new first one among the words with the same initial letter.
2. The process is repeated from the second word of the corrected list.

Any of the two algorithms can be implemented in two variants:

- As a direct pass algorithm: the list is processed from the beginning to the end;
- As a reverse pass algorithm: the list is processed from the end to the beginning.

These implementations give different results.

4.2 Experimental Results

For the experiments, we took Spanish texts on mortgage and crediting. This topic is narrow enough to provide a representative set of similar words. The total number of words considered was 560 (numbers and words with less then 4 letters were excluded); again, only alphabetically adjacent pairs were considered. For the experiments, we used the formula (5). Both direct and reverse pass version of the algorithms were tested. The results proved to be rather similar; Table 4 shows the results for of Algorithm 1.

Table 4. Experimental results.

	Direct pass	Reverse pass
Recall R	92.5%	95.4%
Precision P	89.4%	95.0%
F-measure	90.9%	95.2%

4.3 Discussion

Reverse pass implementation proved gave better results for both algorithms. This is because the length of the common part of the similar words in an alphabetically ordered list increases on average. With a direct pass algorithm, the formula fails to detect similarity of words at the beginning and the end of a group of similar words. A reverse pass algorithm gives strong compensatory effect for truncation of common part of similar words, increasing the probability of their joining.

Algorithm 2 seems to give better results on texts from some narrow domains, where many similar words are separated by others in the lexicographic order and therefore can not be joined by Algorithm 1. However, testing this hypothesis requires more experiments.

5 Conclusions and Future Work

We have suggested a modification of Makagonov's method [7] for testing (morphological) word similarity. His approach is based on an empirical formula trained on a small number of examples. Our proposed modification uses 3-grams instead of letters in this formula. In the paper, we have introduced operations of comparison on *n*-grams, used by the algorithm. Experiments show improvement of the suggested modification over the original algorithm. The suggested modification keeps all properties of the original method: empirical formula does not require any morphological dictionaries of the given language and can be tuned manually (or trained on a small number of examples) on a given language or topic.

In the future, we plan to construct other empirical formulae taking into account statistical regularities of words extracted from the training corpus. We also plan to compare our results using *n*-grams with *n* other than 3.

References

1. Alexandrov, M., Blanco, X., Makagonov, P. (2004). Testing Word Similarity: Language Independent Approach with Examples from Romance. In *Natural Language Processing and Information Systems*, Lecture Notes in Computer Science 3136, Springer, pp. 223-234.
2. Baeza-Yates, R., Ribero-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley.
3. Cramer, H. (1946): *Mathematical methods of statistics*. Cambridge.
4. Gelbukh, A., Sidorov, G. (2002): Morphological Analysis of Inflective Languages through Generation. *Procesamiento de Lenguaje Natural*, No 29, 2002, p. 105–112.
5. Gelbukh, A., Sidorov, G. (2003): Approach to construction of automatic morphological analysis systems for inflective languages with little effort. In: *Computational Linguistics and Intelligent Text Processing* (CICLing-2003), Lecture Notes in Computer Science, Springer, No 2588, pp. 215–220.
6. Ivahnenko, A. (1980): *Manual on typical algorithms of modeling*. Tehnika Publ., Kiev (in Russian).
7. Makagonov, P., Alexandrov, M. (2002): Empirical Formula for Testing Word Similarity and its Application for Constructing a Word Frequency List. In: *Computational Linguistics and Intelligent Text Processing* (CICLing-2002), Lecture Notes in Computer Science, Springer, No 2276, pp. 425–432.
8. Makagonov, P., M. Alexandrov, M., Gelbukh, A. (2004): Formulae for Testing Word Similarity trained on examples. In: *Corpus Linguistics-2004*. Proc. of linguistics seminar of Sankt-Petersburg University, Russia, 15 pp. (in Russian).
9. Porter, M. (1980): An algorithm for suffix stripping. *Program*, 14, pp. 130–137.
10. Renz, I., Ficzay, A., Hitzler, H. (2003): Keyword Extraction for Text Categorization. In: In: *Natural Language Processing and Information Systems*, Lecture Notes in Informatics, GI-Edition Germany, No 129, pp. 228–234.

HMM based POS Tagger for a Relatively Free Word Order Language

Arulmozhi Palanisamy and Sobha Lalitha Devi

AU-KBC Research Centre, MIT Campus,
Chromepet, Chennai - 600044, India.
{p.arulmozhi,sobha}@au-kbc.org

Abstract. We present an implementation of a part-of-speech tagger based on hidden markov model for Tamil, a relatively free word order, morphologically productive and agglutinative language. In HMM we assume that probability of an item in a sequence depends on its immediate predecessor. That is the tag for the current word depends up on the previous word and its tag. Here in the state sequence the tags are considered as states and the transition from one state to another state has a transition probability. The emission probability is the probability of observing a symbol in a particular state. In achieving this, we use viterbi algorithm. The basic tag set including the inflection is 53. Tamil being an agglutinative language, each word has different combinations of tags. Compound words are also used very often. So, the tagset increases to 350, as the combinations become high. The training corpus is tagged with the combination of basic tags and tags for inflection of the word. The evaluation gives encouraging result.

1 Introduction

Part of speech (POS) tagging is the task of labeling each word in a sentence with its appropriate syntactic category called part of speech. It is an essential task for all the language processing activities. There are two factors determining the syntactic category of a word (a) the words lexical probability (e.g. without context, *bank* is more probably a noun than a verb) (b) the words contextual probability (e.g. after a noun or a pronoun, bank is more a verb than a noun, as in "I bank with HSBC "). Hence it disambiguates the part of speech of a word when it occurs in different contexts. For any POS work the tagset of the language has to be developed. It should contain the major tags and the morpho-syntactic features called the sub tags.

In this paper we present a POS tagger developed for Tamil using HMM and Viterbi transition. Tamil is a South Dravidian language, which is relatively free word order, morphologically productive and agglutinative in nature. It is an old language and most of the words are built up from roots by following certain patterns and adding suffixes. Due to the agglutination many combination of words and suffixes occur to form a single word whose POS is a combination of all the suffixed words or suffixes. Hence the total number of tagset increases as the combination increases. In this work we confine to a fixed set of tagset, which gives the most commonly needed tags for any

© *A. Gelbukh (Editor)*
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 37-48

computational purposes. The paper's schema is as follows. The second section discusses about the different techniques used for POS tagging. Third section explains the tagset and the architecture of the system developed and the fourth section discusses the implementation details. The results are discussed in the next section (5) and the sixth section concludes the paper.

2 Tagging Techniques

Many techniques have been used for developing POS taggers. The different techniques are (1) Rule-based (2) Transformation Based and (3) statistical based. Each technique has its own merits and demerits.

The first technique to be developed was the rule-based tagger. These taggers used context sensitive rules in getting the correct tag. Among the rule-based approaches the one that achieved the best accuracy is 97.5%[1]. These taggers are based on a defined set of hand written rules. Most of the existing Rule Based POS taggers are based on two-stage architecture. The first stage assigns a list of probable tags (or the basic tag) for a particular word. The second stage uses the list of disambiguation rules, to reduce the list (or change a wrong tag) to a single right tag. The Brill's tagger initially tags a sentence by assigning each word its most likely tag, and then that is tagged according to context information at the second stage [2]. Here all the rules are pre-defined. The rules can be broadly classified in to two (1) Lexical Rules and (2) Context Sensitive Rules. The lexical rules act at the word level. The context sensitive rules act at the sentence level.

The rule-based taggers are developed for European languages. In Indian languages a rule-based POS tagger for Tamil was developed and tested [3]. It uses a suffix stripper to get the root word and no dictionary is used. The tagger tags the major tags (N, V, etc) of the root word. For Example, The word patikkira:n "reading+m+sng+3p" can be split in to pati "Read" V + kkir Pres + a:n Sng.M.3P. The tag for the root word pati cannot be identified without a dictionary. But from the suffixes it takes it could be identified whether it is a noun or a verb.. Here in this example, using the tense marker and GNP marker, the root word can be assumed as a verb. This is done in the word level. If there is any ambiguity, the word is left unknown. And it is resolved in the next step – context level. There are nearly 90 lexical rules and nearly 7 context sensitive rules, which are hand crafted. The precision of the system is 92%. This system gives only the major tags and the sub tags are overlooked while evaluation. Tagging becomes difficult when there is no inflection in the word. When the sub tags are considered the performance is reduced.

Transformation Based Learning method is an approach in which the rule-based and the stochastic methods are combined and used. Like the rule-based taggers, this is based on rules, which say what tag to be assigned to what words. And like the stochastic taggers, this is a (supervised) machine learning technique, in which the rules are automatically derived from the (pre tagged) training data. And those rules are applied to the test corpus. There are various algorithms, which are used for learning purpose. The most commonly used Brill's tagger [4, 5] (for English) is a TBL based tagger. Brill's algorithm has three major stages. In first stage, it labels

every word with the most likely tag. In second stage, it examines all the possible transformations and selects the one, which gives the most improved tagging. The stage two requires that the TBL algorithm knows the correct tag for each word. That is the TBL is a supervised learning algorithm [1]. The output of a TBL process is a list of transformations. The transformation consists of two parts: A triggering environment and a rewrite rule. Then the third stage - it re-tags the data according to the rule. These three steps are repeated till some threshold condition reaches. The accuracy of the tagger depends upon the training corpus and the set of rules. Apart from Brill's tagger, the fn-TBL [6] toolkit and mu-tbl [7] tool uses TBL for tagging English sentences. These are trained for English and also for other European languages.

The POS taggers are developed using statistical method. This is based on probability measures. Generally this needs a tagged corpus for training. The training corpus is a tagged corpus, which is assumed to be 100% correct. The probabilities are calculated using unigram, bigram, trigram, and n-gram methods [8]. Unigram Probability is the probability of a word to occur in a corpus. The Bigram model calculates the probability of a word, given the previous word. By definition, the bigram model approximates the probability of a word given all the previous words, by the conditional probability of the preceding word. Here an assumption is made that the probability of a word depends only on the previous word. This is called Markov assumption. Bigram model is called the first order Markov model, trigram is called the second order Markov model and an N-gram model is called the N-1th Markov model. In developing a POS tagger, the tags are called states in the Markov model and the the words are called the symbols. When N=2, this is generally called a Hidden Markov Model as the information about the previous states are hidden. Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of hidden states, known as the Viterbi path.

For many morphologically rich languages like Japanese and Arabic, viterbi algorithm is used for tagging and disambiguation [9, 10]. The Arabic POS tagger achieved an accuracy of around 90% when disambiguating ambiguous words. But unknown words are tagged as nouns because the tagged training corpus had 67% of nouns [10]. Also for languages like Chinese and Japanese the word segmentation itself becomes a problem because of the nature of the language. Still Chinese part of-speech tagging using a first-order fully connected hidden Markov model gives promising results [11].

Regardless of whether one is using HMMs, maximum entropy conditional sequence models, or other techniques like decision trees, most systems work in one direction through the sequence (normally left to right, but occasionally right to left). Most of the well-known and most successful approaches are unidirectional [12].

2.1 Tamil Tagset

As Tamil is an agglutinative language, each word has different combinations of tags. Here we have 17 basic tags and 31 sub tags. Compound words are very common in this language. So the word will have the combination of basic tags and tags for inflection. Thus we have 350 unique tagset for the training data. The nouns take the

case markers and plural markers and verbs take perfect, imperative, person number, gender and tense markings. Other than this, pronouns have person number and gender and also the clitics which are suffixed to the root word to form adverbs, conjunctions etc. The dates, numbers and punctuations are also tagged separately. The tags we use are not monadic tags but structured tags such as <N+Pl+ACC>. This representation is required for further syntactic analysis because different pieces of information in the tag serve different roles in the syntactic representation. The representation is given below:

avan	*kataikku*	*cenRu*	*maruntukaLai*	*va:nkina:n*
He	shop (DAT)	go (VBP)	medicine (PL+ACC)	buy (PST+3SM)

avan/PR+3SM *kataikku*/N+DAT cenRu/V+VBP *maruntukaLai*/N+PL+ACC va:nkina:n/ V+PST+3SM

(He went to the shop and bought medicines.)

The main tags and the sub tags, which can occur with the main tag, are given in Tables 1 and 2.

Table 1. Main tags.

	Main Tag	Representation
1	Noun	N
2	Verb	V
3	Adjective	ADJ
4	Adverb	ADV
5	Verbal Participle	VBP
6	Relative Participle	AJP
7	Quantifier	Q
8	Conjunction	CNJ
9	Punctuation	PNC
10	Indeclinable	IND
11	Interrogative	INT
12	Ordinal	ORD
13	Pronoun	PR
14	Verbal Noun	VBN
15	Determiner	DET
16	Symbol	SYM
17	Unknown	UNK

2.2 Agglutinating Nature of the Language

In this section, we give a detailed description of the agglutinating nature of Tamil, and how difficult is to analyze the inflected/compound words. The grammatical properties of Tamil comprise both morphological and syntactic categories. Generally the main parts of speech in modern Tamil can be distinguished into three morphological word classes as Nouns, Verbs and Uninflected words.

Table 2. Sub tags

	Sub tag	Representation		Sub tag	Representation
1	2nd person, Plural, Neuter gender	2PN	12	Clitic	CLT
			13	Compound	COMP
2	2nd person, Singular, Honorific	2SH	14	Conditional	CND
			15	Dative	DAT
3	2nd person, Singular, Masculine	2SM	16	Emphatic	EMP
			17	Future Tense	FUT
4	2nd person, Singular, Neuter gender	2SN	18	Genitive	GEN
			19	Hortative	HRT
5	3rd person, Plural, Neuter gender	3PN	20	Imperative	IMP
			21	Inclusive	INC
6	3rd person, Singular, Feminine	3SF	22	Infinitive	INF
			23	Instrumental	INS
7	3rd person, Singular, Honorific	3SH	24	Locative	LOC
			25	Negative	NEG
8	3rd person, Singular, Masculine	3SM	26	Past Tense	PST
			27	Plural	PL
9	3rd person, Singular, Neuter Gender	3SN	28	Post Position	PSP
			29	Present Tense	PRE
10	Ablative	ABL	30	Representative	REP
11	Accusative	ACC	31	Sociative	SOC

Words in Tamil can be segmentized in to a sequence of parts called morphemes. Thus Tamil morphology can be characterized as agglutinating or concatenative [13]. This can be represented as follows.

$$\text{Stem } (+ \text{ affix})^n$$

Where the superscript n means one or more occurrences of suffixes. When morphs are serialized as suffixes at the right end of a word stem, inflected or derived words are formed. For example, an inflected noun consists of a noun stem followed by a plural suffix and/or a case suffix:

"*vItukaLil*" = *vItu* + *kaL* + *il* = house + PL + LOC = " in the houses ".

An inflected verb consists of a verb stem followed by a tense and PNG marker.

"*vantAn*" = *va* + *nt* + *An* = come + PST + 3SM = " he came ".

Apart from the inflection of stems, there can be oblique stems occurring with case suffixes, postpositions etc. when words are concatenating, to form a single compound word, almost all the combinations are valid. The most common rules for concatenation are, Noun stem + head noun, oblique stem + case suffix, Noun + Verb, Verb + Verb, Determiner + Noun, Adverb + Verb, Derived noun, Derived verb, {Noun, Verb, Adjective, Adverb} + Postposition, verbal participle+ verb, Adjectival participle + Postposition, Adverb + Verb + Postposition etc. Few examples are given below:

Rule 1: **Noun stem + head noun**
talaimuti = *talai* + *muti* = head hair = "hair of the head"

Rule 2: **oblique stem + case suffix**
enakku = *nAn (obl)* + *ku* = I (obl) + dat = " to me"

Rule 3: **Noun + Verb**
kaivaikAte = *kai* + *vai*+*Ate* = hand + put + (neg) = "do not put (your) hand"

Rule 4: **Verb + Verb**
cAppitamutiyum = *cAppita* + *mutiyum* = eat + can do = "can eat"

Depending on the position of suffixes, when it is added to words, same suffix may give different meaning at different places. For example, the suffix "um" when added to a noun, it is an inclusive marker. When it is added to verb, that indicates third person singular neuter gender. Another example would be postpositions. They can stand-alone, add as a suffix of a word or it can occur between two words as a glue and make a compound word. Few examples are given below

"vantapintAnenRu" = vanta + pin + tAn + enRu = "only after (somebody) came" = AJP + PSP + EMP + IND

"kARRuvazhicceyti" = kARRu + vazhi + ceyti = "news by word of mouth" = N + PSP + N

"enakkumun" = enakku + mun = "before me" = N(DAT) + PSP

"polave" = pola + e = "like that" = PSP + EMP

Thus, the concatenative nature of the language makes the POS tagging task more difficult. There are number of words which have the same spelling and act as different part of speech in a sentence.

Example:

iru – acts as a quantifier (two) and as a verb (be there)
pitikkum – acts as an RP and as a verb.
moodi – acts as a noun and verb.

Tamil is a relatively free word order language where the constituents in a sentence could be rearranged in all possible combinations without altering the meaning. The following example shows this nature of Tamil.

Example:

"He hit the ball with a bat" can be written in different ways,

a. *avan pantai mattaiAl atittAn*
b. *pantai avan mattaiAl atittAn*
c. *avan mattaiAl pantai atittAn*
d. *mattaiAl avan pantai atittAn*
e. * *avan atittAn pantai mattaiAl*

The sentence "e" is not a grammatical sentence in Tamil, since it is not verb ending. Handling of the tagging process for Tamil is explained in the next section.

3 The POS Tagger

The system has two major modules, where the first module is for building the language model and the second for finding out the viterbi tag sequence for a given sentence. The training corpus is assumed to be 100% correct. The training corpus consists of 25000 tagged words, which are tagged by a rule-based tagger and then manually checked and corrected.

The training corpus is given to the training module and the initial matrices for the viterbi algorithm are built. There are 3 matrices built by this module – initial state probability matrix (π), transition probability matrix (A) and emission probability matrix (B). Then a language model (λ) is built using the formula,

$$\lambda = (A, B, \pi)$$

The second module is the actual viterbi algorithm, which finds out the most likely tag sequence for a given sentence. The input sentence is given to this module and the output is the tagged sentence. How viterbi algorithm works for tagging a sentence is explained as follows.

Probability of item in sequence depends on its immediate predecessor. That is the tag for the current word depends up on the previous word and its tag. Here in the state sequence the tags are considered as states and the transition from one state to another state has a transition probability. The emission probability is the probability of observing a symbol in a particular state. Here in the POS tagger application words are considered as symbols and the tags are considered as states.

Training of the system using tagged Tamil corpus and using it for tagging a new unseen sentence is explained in section 4. As Tamil is an agglutinative language, each word may have different combinations of tags. Compound words are also used very often. The training corpus is tagged with the basic tag and the tags for the inflection of the word. Implementation of the modules and tagging a new corpus using the system are explained in detail in the following sub sections.

3.1 System Implementation

The system is implemented using the concept of HMM and Viterbi transitions. Hidden Markov Model (HMM) is a statistical method, which can be used for Natural Language Processing Tasks like Parts of Speech tagger, Information Extraction etc. Using HMM an attempt is made to build a Language model, such that the system (or the computer) can be trained to work similar to humans for processing natural languages like English, Tamil etc.

There are some basic assumptions made when HMM is modelled for natural language texts. First assumption is that Language is a Markov Process with unknown parameters or hidden states and we have to determine the hidden parameters with the use of observable output parameters. In a Markov model there are no hidden parameters all the parameters are observable.

There are 3 canonical problems to solve with HMMs:

1) Given the model parameters, compute the probability of a particular output sequence. Solved by the *forward algorithm.*
2) Given the model parameters, find the most likely sequence of (hidden) states which could have generated a given output sequence. Solved by the *Viterbi algorithm.*
3) Given an output sequence, find the most likely set of state transition and output probabilities. Solved by the *Baum-Welch algorithm.*

In constructing HMM for language model, we have to determine what are the states or parameters in the model and what they correspond to and how many total numbers of states are possible in the model. Prior to building of HMM, a state table has to be constructed. Once the state table is prepared and annotated training text is available we can build HMM model.

The HMM model (λ) is given by

$$\lambda = (A, B, \pi) \tag{1}$$

Here A, B, and π are the probability measures used for modeling. Let us say there are N, distinct states and states denoted as $S_1, S_2, \ldots, S_N$. And M number of distinct output symbols per state and denoted by $V_1, V_2 \ldots V_M$. If we denote a state at time t as q_t then A is the state transition probability distribution.

$$a_{ij} = P\,[q_{t+1} \mid q_t] \tag{2}$$

B is the observation symbol probability distribution in state j, $B=\{b_j(k)\}$ where

$$b_j(k) = P\,[\,V_k \text{ at } t \mid qt = Sj] \qquad \begin{matrix} 1 < j < N \\ 1 < k < M \end{matrix} \tag{3}$$

π is the initial state distribution where

$$\pi_i = P\,[\,q_1 = S_i\,] \qquad 1 < i < N \tag{4}$$

After calculating the initial state distribution, Viterbi algorithm can be implemented for finding out the hidden sequence of states for a particular input sentence. In this algorithm the following assumptions are made. First, both the observed events and hidden events must be in a sequence. This sequence often corresponds to time. Second, these two sequences need to be aligned, and an observed event needs to correspond to exactly one hidden event. Third, computing the most likely hidden sequence up to a certain point t must depend only on the observed event at point t, and the most likely sequence at point t − 1. These assumptions are all satisfied in a first-order hidden Markov model. What the algorithm does only is it tries to give the best possible solution. The difficulty lies in setting the optimal criteria for choosing the path. There are several possible optimal criteria's, which can be set. In Viterbi algorithm a single best state sequence is given as output. Here we find the path that maximizes the observations given the HMM model.

The formal algorithm consists of 5 steps. First step initializes the first column using the initial state vector and the first observation. Next is the recursion step, which gives the i-th element of the t-th column, which is the probability of the *most likely* way of being in that position, given events at time t–1. The back pointer indicates where one is most likely to have come from at time t–1 if currently in state i at time t. next step is

termination. It indicates what the most likely state is at time T, given the preceding T-1 states and the observations. Fifth step traces the back pointers through the lattice, initializing from the most likely final state. Using backtracking, we find the most likely tag sequence determined by the Viterbi algorithm. How a new sentence assigned the most likely tag sequence is explained below.

Here we assume each tag of POS as a state. The words in the text are assumed to be as observable symbols or tokens. If there is large enough text corpus, which contains of all possible sentence structures of the language then using HMM we can build a very robust language model for this task.

The initial state probability distribution (π) is constructed by finding the ratio of the count of each states occurring in the first position of the sentence to the total number of sentences. For example if state S1 has occurred n times as the first state of sentence in the corpus and if there are X number of sentences in the corpus then initial state probability for state S1 is given as (n/X). We similarly find for all states in the model and an initial state probability distribution is built.

The transition state probability distribution is nothing but the states bigram frequency. Or we can say it is the probability of state Si to transit to state Sj.

The observation symbol probability distribution or simply called emission probability is the probability of an output symbol Vk to occur in given particular state Si. That is probability for a word to have a particular tag.

Once these three (A, B, π) model parameters are found we use Viterbi transitions to find the best path for a given sequence of output symbols (i.e., a test sentence).

The training corpus is tagged by a rule-based system [3] and manually corrected. The tag for each word is a combination of basic tags and the tag for inflection.

Example 1:

the word *payanpatukinRana* - "they are useful" is tagged as "V+PRE+3PN".
That is, this word is formed with, "*payanpatu+kinRu+ana*"
payanpatu "are useful" – Verb (V)
kinRu – Present tense (PRE)
ana – 3rd person, Plural, Neuter Gender (3PN)
The final output of the tagger is V+PRE+3PN

Example 2:

The word *immaruntukaL* "the medicines" - is tagged as "N+COMP"
This compound word is formed with, "i+maruntu+kaL"
i (inta) "this" – Determiner (DET)
maruntu "medicine" – Noun (N)
kaL – Plural (PL)

When the combination of the basic tag increases, the tagset for viterbi increases. This leads to sparse data problem [14]. To avoid this problem, we reduce certain combination of tags to a single tag "COMP".

Using the training corpus, the initial state matrix, transition and emission probability matrices are calculated and the language model is built. There are many paths in an HMM with S states. We try to find the path that maximizes the observations given the HMM model [15].

As the corpus is initially tagged using a rule-based system and corrected, it takes care of the inflection to some level reducing the manual work, though compounding is not dealt in detail.

4 Results and discussion

The system is tested against a data set of 5000 words. The system performs well and gives a high precision and recall. Three different sets of test data are tagged and evaluated. These sets are from different domains given below.

Set 1 – Recipes
Set 2 – Veterinary Diseases
Set 3 – Historical Events

Three different types of data are taken because the constituents in a sentence are different in different domains. In recipes, multiple proper nouns occur continuously, separated by a punctuation mark. They are written in a step-by-step manner and most of the sentences are imperative. Whereas in veterinary diseases, there are considerable number of foreign words, transliterated medicine names etc and the format in which the text is written, mostly resembles textbooks. In historical domain, the paragraphs are highly related to each other and it is written like a story. Here, we come across many classical words from ancient Tamil. There are many compound words. Three sets of data containing 1565, 1810 and 1616 words are taken from the above domains for evaluation and the precision and recall is calculated. The precision for the 3 sets is 98.43% and the recall is 98.43%. The evaluation results are given in Table 1.

Table 3. Evaluation Results

Title	No. Of Words	Words tagged	Correctly Tagged	Precision	Recall
Set 1	1565	1565	1544	98.7%	98.7%
Set 2	1810	1810	1779	98.3%	98.3%
Set 3	1616	1616	1589	98.3%	98.3%

An advantage in using viterbi algorithm is the system is not influenced by small errors. For example,

enna – "what"

Here, the actual tag for this is "INT" but in the manually tagged corpus, it was wrongly tagged as "IND" at one place. But that is not carried to the output. Even if there is a small typological error in the tagged corpus, which changes the tag itself, the machine tagged output is correct. Even if there is an unseen word, the system tries to find out the tag for that word depending on the context. This does not need an exhaustive set of rules. As the training corpus increases, the system becomes robust.

As Tamil is an agglutinative language, each word may have different combinations of tags. Compound words are also used very often. The training corpus is tagged with the combination of basic tags and tags for inflection of the word. So, the tagset

increases to 350, as the combinations become high. The basic tag set including the inflection is 53.

If any unseen word (which does not occur in the training corpus) occurs in the test corpus, the emission probability of the word becomes zero. But when viterbi algorithm is implemented, it tries to assume the tag for the unseen word depending on the previous sequence of words. It explores all possible states for emitting the word. If no tags are occurring after a particular tag, transition probability of that particular tag becomes zero and there are no probable states after that. In that case, the whole sentence is not tagged because the total probability becomes zero. When there is a very small probability especially for special characters, the transition probability becomes insignificant and it becomes zero, leading to the total probability to be zero. When the insignificant output symbols such as ";", "!", etc. occurs in between a sentence, the above-mentioned problem is faced.

Some of the sentences are not tagged because at some point for an unseen word, no transition may be possible from the current state. So, no tags are assigned. Some words are assigned wrong tags, because transition or emission probability is wrongly calculated. This is because some forms of words do not occur in the training corpus and the test corpus has those forms. So, there is a high probability of assigning a wrong tag to those particular words. For example,

a. the word *" eRiya"* which means, "to throw" is tagged as "ADJ" by the machine. But the actual tag for this is "ADV".

b. the word *irukkum* which means "will be there" is tagged as V+FUT+3SN. But the actual tag for this in that particular sentence is "V+AJP".

5 Conclusion

In this paper we have discussed a statistical part of speech tagger for Tamil which is developed as a module in the Tamil- English Machine Translation system. In arriving at the POS tagger we use HMM and Viterbi algorithm for tagging a new corpus. The system performs well with high precision and recall. HMM emission probability for an unknown word is calculated depending on the word order. As Tamil is a relatively free word order language and morphologically rich, there is always a high probability for an unseen word to occur. So the precision may vary according to the training corpus. The system can be made more reliable and robust when trained with a bigger corpus, which contains all types of words and its combinations equally distributed. A high precision morphological analyzer is not needed in HMM; instead a compound word analyzer will improve the performance considerably. Another advantage in this system is even if there is any man made mistake in the training corpus, that is not carried to the output. This POS tagging task for Tamil is useful and it plays a vital role in language processing activities like information extraction, machine translation etc. from Tamil to other languages.

References

1. Eric Brill: Some Advances in transformation Based Part of Speech Tagging, In proceedings of the Twelfth International Conference on Artificial Intelligence (AAAI-94), Seattle, WA, (1994)
2. Eric Brill: A Simple Rule-Based Parts of Speech Tagger, University of Pennsylvania,(1992)
3. Arulmozhi. P, Sobha. L, Kumara Shanmugam. B.: Parts of Speech Tagger for Tamil, Symposium on Indian Morphology, Phonology & Language Engineering, Indian Institute of Technology, Kharagpur (2004), 55-57
4. Eric Brill: Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging, Association of Computational Linguistics, (1995)
5. Eric Brill: Transformation-Based Part of Speech Tagger (V 1.14), http://www.cs.jhu.edu/~brill/code.html
6. Radu Florian and Grace Ngai: Fast Transformation-Based Learning Toolkit, http://nlp.cs.jhu.edu/~rflorian/fntbl/ Johns Hopkins University,
7. Torbjörn Lager: The μ-TBL system, Paper presented at the Third International Workshop on Computational Natural Language Learning (CoNLL'99), Bergen, (1999)
8. Eugene Charniak: Statistical Language Learning,. MIT Press, Cambridge, London (1997)
9. Jun'ichi Kazama, Yusuke Miyao and Jun'ichi Tsujii "A Maximum Entropy Tagger with Unsupervised Hidden Markov Models". In Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium NLPRS, (2001), 333—340
10. Shereen KHOJA: APT: Arabic Part-of-speech Tagger, Proceedings of the Student Workshop at the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL2001), Carnegie Mellon University, Pittsburgh, Pennsylvania, (2001)
11. Chao-Huang Chang and Cheng-Der Chen: HMM-based Part-of-Speech Tagging for Chinese Corpora, Industrial Technology Research Institute Chutung, Hsinchu 31015, Taiwan, R.O.C
12. Kristina Toutanova *et al.*: Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network, HLT-NAACL, (2003), 252-259
13. Thomas Lehmann: A Grammar of Modern Tamil, Pondicherry institute of Linguistics and culture, Pondicherry, India, (1993)
14. Wolfgang Lezius: Morphy - German Morphology, Part-of-Speech Tagging and Applications, ; Stefan Evert; Egbert Lehmann and Christian Rohrer, editors, Proceedings of the 9th EURALEX International Congress, (2000), 619-623
15. Trevor Cohen, Franklin Din, Karina Tulipano: Hidden Markov Models – Methods in Biomedical Informatics. A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", Proceedings of the IEEE, Vol. 77, No 2. (1989)

Extended Tagging in Requirements Engineering

Günther Fliedl, Christian Kop, Heinrich C. Mayr,
Jürgen Vöhringer, Georg Weber, Christian Winkler

University of Klagenfurt

Abstract. In this paper standard tagging mechanisms are discussed and partly criticized. We propose a semantically and morphosyntactically enriched mechanism in which many of the shortcomings of standard POS-taggers are eliminated. Therefore rule based disambiguation steps are presented. They include mainly a specification of contextually motivated verbal subcategories. We need this fine grained system for better preprocessing of requirements texts which have to be highly explicit and non ambiguous. This linguistic preprocessor can support an interpretation tool used to extract semantic concepts and relations for the requirements engineering step in software development.

1 Introduction

Classical tagging approaches use standardized (POS) tag sets. Such kind of standardized tagging (e.g., Brilltagger [1], TnT [2], Q-Tag [11] Treetagger [8], [10] etc.), however, show weakness in the following three aspects:

- Tags like 'VAINF' provide only basic categorial and morphological information;
- Ambiguity cannot be made explicit;
- Chunking and identification of multiple tokens is not possible.

To avoid such deficiencies, we developed a system called NIBA[1]-Tag which allows tagging of German with an extended tagset, and inheritance of morphosyntactic and morphosemantic features. Morphosemantic tagging in our sense is labeling words by morpho-syntactically relevant semantic classifiers (sem-tags like 'tvag2', 'eV', 'indef', 'poss', etc.; see Appendix 1 for a rough comparison with the Treebank [7] STTS [S*99]), It has proved to be an efficient method for extracting different types of linguistically motivated information coded in text. The XML-Tagger-output for the German PP (=P2) ***Bei Eintreffen des Auftrags*** in Table 1 shows how the tagging result is structured.

As can be seen in Table 1, our tagging system has the following linguistic competence:

[1] NIBA is a German acronym for Natural language based Requirements Engineering. The project NIBA is supported by the "Klaus Tschira Stiftung Heidelberg".

© *A. Gelbukh (Editor)*
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 49-56

Table 1. Categories and features generated by NIBA-TAG

Bei (on)	*Eintreffen (arriving)*	*eines (of-the)*	*Auftrags (order)*
lowerCase="**bei**"	lowerCase="**eintreffen**"	lowerCase="**eines**"	lowerCase="**auftrags**"
id="**100713**"	id="**352487**"	id="**976515**"	id="**413424**"
Category = **p0**	Category = **v0**	Category = **det0**	Category = **n0**
type="**temp**"	referTo="**eintreffen**"	type="**temp**"	referTo="**Auftrag**"
	referid="**8264**"		referid="**18541**"
	base-form="**eintreffen**"		base-form="**Auftrag**"
	ps3_sg_praes_ind="**trifft**"	num="**sg**"	numerus="**sg**"
		kas="**gen**"	kasus="**gen**"
		gen="**masc**"	genus="**masc**"
	partikel="**ein**"		
	verbclass="**eV**"		
	verbclass-number="**2**"		
	pp="**eingetroffen**"		

1. The assignment of POS-tags enriched with morpho-syntactic features (e.g., Category = p0/v0/n0);
2. Morpho-semantically motivated subclass tags for verbs (e.g., verbclass = eV)
3. Identification of unknown words through assignment of affix-rules (e.g., kas = gen of *Auftrags*);
4. Extended lemmatizing (e.g., referTo = *eintreffen*).

This competence is used in the requirements engineering process. The rest of the paper is structured as follows. In chapter 2 some verb classes with high frequency are discussed. In chapter 3 we will represent the tagging rules. In chapter 4 we argue for a multilevel interpretation process during requirements engineering.

2 Verbclasses Used by NIBA-Tag

Since a merely morphological classification of verbtags as commonly practised by most taggers is not suitable for many tasks in the field of requirements engineering, a subclassification of the respective tags with respect to the verb arguments is necessary.

In our system German verbs are categorized with respect to the 26 verb classes, which are divided into 12 main-verb classes [3]. In the following we list definitions and examples of the top six most frequent tags for verb classes:

Verbclass 2 – **eV**: Ergative Verbs trigger non-agentive subjects (e.g., *sterben – to die*).

Verbclass 3 – **iV**: Intransitive verbs which need only one argument (the subject) (e.g., *schlafen – to sleep*)

Verbclass 6 – **psychV**: bivalent verbs with a psychologically motivated goal argument. No passivation is possible for that kind of verbs. (e.g., *ärgern – to annoy*)

Verbclass 7 – **tVag/2** verbs select two arguments: an agent role and a thema role. These verbs allow passivation; (e.g., *lesen – to read*).

Verbclass 7.2 – **tVag2pp**: Transitive Verbs with prepositional object. (e.g., *hineinschneiden – cut into sth.*)

Verbclass 8 – **tV3**: Ditransitive Verbs with an agentive subject and 2 objects (e.g., *Der Kollege gab mir einen Rat - the colleague gave me an advice*).

Verbclass 11 – **tV2**: Bivalent verbs with a non agentive subject and a thematic object (e.g., *Der Schwamm absorbiert die Flüssigkeit – the sponge absorbs the liquid).*

3 Substantial Characteristics of the Tagging System

The system can be characterized through the following features and components [4]:

- A fine grained categorization system for open and closed German word classes. Lexicon categories are subdivided into semantically motivated subclasses, labelled with attributes. See Appendix 1;
- A fully functional lexical database with an integrated rule component for the optional generation of full German word forms;
- Export – and import functionality providing the import of simple structured excel sheets and export to XML, which can be converted to a tagger specific lexicon LeXML, a Berkeley DB);
- An extended rule based Perl tagger, including different types of context rules.

Subsequently, much effort had to be spent in the definition of context rules facilitating the disambiguation during the tagging process [5].

The following rule-types have been developed:

1. Feature filtering and generation rules;
2. Categorization rules;
3. Conversion rules;
4. Disambiguation and deletion rules;
5. Priorisation rules;
6. Chunk rules.

Feature filtering and generation rules Many features of lexical units are context-dependent; e.g.,

Bei[loc] → *Bei* [temp]/[Verbal noun]
e.g., *Bei Eintreffen des Auftrags (on arrival of the order)*
[Aux0] → [pass]/[past participle of tvag2, tv3]

Categorization rules Categorization rules are part of the lexicon component, initialized through inflectional expansion of basic word forms for the purpose of generating paradigms. NTMS based part-of-speech tags divide words into morphosyntactically and semantically motivated categories, based on how they can be combined to build up sentences.

Conversion rules Conversion rules are typical local context rules. They operate on lexically generated categorizations to transform them into different word classes, e.g., *eintreffen* (*to arrive*) v0 [eV]-> n0. In the XML-output, the original categorial grid remains visible.

Disambiguation and deletion rules They delete contextually not acceptable attribute values, e.g., the accusative in the context of '*des*'. Attributes which cannot be disambiguated, are being deleted likewise.

Priorisation rules During the tagging process attribute values are counted, whereas those with high frequency are priorized.

Chunk-rules Chunk rules refer to the results of the basic tagging process building phrasal categories from lexical categories, mainly N3 (NPs) and P2 (PPs). NPs are identified as expansions of nouns.

4 The NIBA Tag within Requirements Engineering

Niba Tag is currently used in the Requirements Engineering Project NIBA. In Software Development, Requirements Engineering is an important phase with the objective to find out the structural, functional and dynamic aspects of the software [9]. Extracting of functional, structural and behavioural aspects is thus one of the main tasks during this step. For the purpose of modeling requirements a modeling language called KCPM (Klagenfurt Conceptual Predesign Model) with a small set of modeling notions (*thing type, connection types, operation types, conditions*) is used to describe aspects of software. The notions are presented in glossaries. Thing types and connection types cover the structural aspects of software development. Operation types and conditions focus on the functional and behavioural aspect [6].

To extract these concepts out of the tagging results, two additional tools are necessary. A NIBA Dynamic Interpreter and a NIBA Static Interpreter were developed. This paper focuses on the Dynamic Interpreter since it also covers some structural aspects.

Before the interpreter can extract the concept it has to do some core linguistic work. The interpreter must assign syntactic functions and semantic roles to those word groups which are identified as syntactic phrases by NIBA-TAG. This is quite a difficult task for German sentences because of (morpho-) syntactic problems in German (free word order etc.). The interpreter makes linguistically motivated default decisions based on verb class related predicate argument structures (see Section 2). The assignments of the syntactic functions and semantic roles are then used for the interpretation process. The results are shown in the right upper and lower part of the window. The interpretation process presupposes decisions about the interpretability of the tagged sentences. Currently a first and simple model of four levels (level 0 – level 3) was introduced:

Level 0: No interpretation of the sentence is possible at all. This means that the sentence is written in such a way that the interpreter cannot find any structure within the sentence which can be used for interpretation (see level 1).

Level 1: At least one of the two parts *implication* and/or *condition* is found.

Level 2: Verbs are found, but cannot be associated with arguments.

Level 3: Candidates for verb arguments are recognized for the verb (if there is only one in the sentence) or for at least one verb (if there are more of them in the sentence).

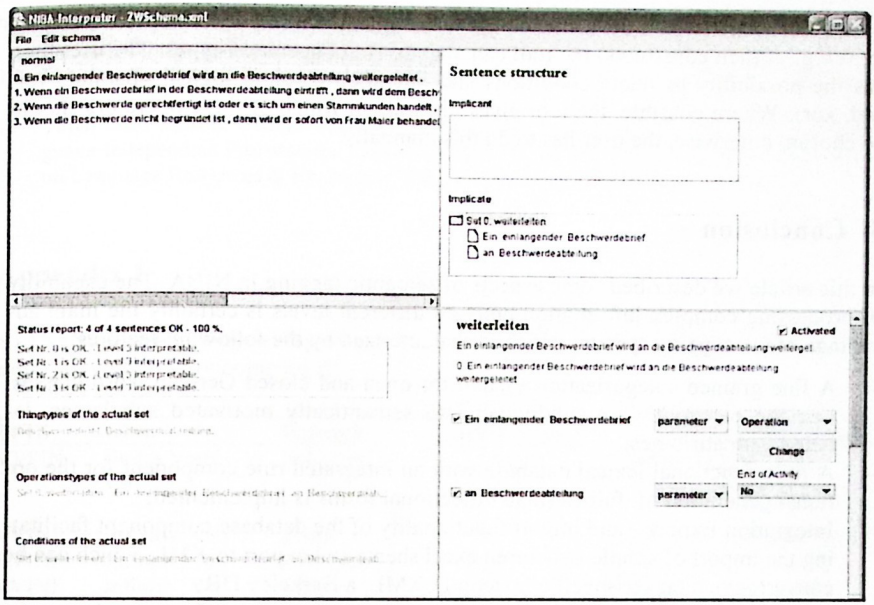


Fig. 1 Main window of the interpreter

If a sentence can be interpreted on level 3, it is possible to derive KCPM notions. This can be controlled in the right lower corner of the window. For each verb and its arguments in the implication section (upper right part) the end user receives a default interpretation. If the verb is an agentive verb then it is mapped to an operation-type. The noun which is the syntactic subject of the sentence is mapped to the acting actor. The nouns which could be the syntactic objects are mapped to parameters. If the verb is not an agentive verb then it is interpreted as a condition. All the arguments of the verbs in the sentence are listed as candidates for involved thing-types. Those arguments which are possible thing types are filtered out based on a default filtering mechanism. If the sentence "*a person owns a car*" is taken as a condition, then both "*person*" and "*car*" are candidates for the involved thing-type but only one of these nouns is chosen as an involved thing-type (*person*). The rest of the sentence "*owns a car*" is then treated as the property of that thing-type necessary to fulfill a condition.

All interpreter results are understood to be "default", i.e., the user can always overrule default decisions. For example, the user can change the type of each thing-type (parameter, calling actor, acting actor) if he thinks that the default assumption is not appropriate.

Furthermore, the tool currently distinguishes between sentences (sentence parts) that are useful for interpretation and sentences which are not ("*fill-ins*"). In fact, the tool assumes that every sentence should be interpreted. However, there is a check box "Activated" which is "on" by default. If the user disables this check box, then the interpretation result will not be transferred into in the final schema.

In some cases (where the sentence fits with some given implicational sentence patterns e.g., if/then constructs) the tool can also derive cooperation-types. The user then has the possibility to relate conditions and operation-types to logical operators (or, and, xor). Where possible, the tool gives hints about which of these operators should be chosen, otherwise, the user has to do this manually.

5 Conclusion

In this article we described some aspects of semantic tagging in NIBA. The capability of processing complex information units on different levels is certainly the main advantage of our tagging system, which is characterized by the following features:

- A fine grained categorization system for open and closed German word classes. Lexicon categories are subdivided into semantically motivated subclasses, labeled with attributes;
- A fully functional lexical database with an integrated rule component for the optional generation of full German inflectional forms is implemented;
- Integration Export – and import functionality of the database component facilitating the import of simple structured excel sheets and export to XML, which can be converted to a tagger specific lexicon LeXML, a Berkeley DB;
- The linguistic tasks done by the tools presented above are a first step during the computer supported extraction process of concepts for software development.

References

1. Brill, E.: A simple rule-based part of speech tagger In: Proceedings of the Third Conference on Applied Natural Language Processing, ACL, 1992.
2. Brants, T.: TnT - A Statistical Part-of-Speech Tagger. In: Proc. of the 6th Applied Natural Language Processing Conference ANLP-2000. Seattle, pp. 224–231
3. Fliedl, G.: Natürlichkeitstheoretische Morphosyntax. Aspekte der Theorie und Implementierung. Habilitationsschrift. Gunter Narr Verlag. Tübingen. 1999.
4. Fliedl, G.; Kop, Ch.; Mayerthaler, W.; Mayr, H.C.; Winkler, Ch.; Weber, G.; Salbrechter, A.: Semantic Tagging and Chunk-Parsing in Dynamic Modeling. In: Meziane F.; Metais E.: (eds.) Proceedings of the 9th International Conference on Applications of Natural Language Processing and Information Systems, NLDB2004, Salford UK, Springer LNCS 3316 pp. 421 – 426.
5. Fliedl, G., Weber, G.: Niba-Tag - A Tool for Analyzing and Preparing German Texts. In: Zanasi, A.; Brebbia C.A.; Ebecken, N.F.F.; Melli, P. (Ed.): Data Mining 2002 Bologna: Wittpress September 2002 (Management Information Systems, Vol 6), pp. 331–337.
6. Kop C., Mayr H.C.: An Interlingua based Approach to Derive State Charts form Natural Language Requirements In: Hamza M.H. (Ed.): Proceedings of 7th IASTED International Conference on Software Engineering and Applications, Acta Press, 2003, pp. 538–543.
7. Marcus, M.; Santorini, B. and Marcienkiewicz, M.: Building a large annotated corpus of English: the Penn Treebank., Computational Linguistics, 1993.
8. Schmid, H.: Probabilistic Part-of-Speech Tagging using Decision Trees. www.ims.uni-stuttgart.de/ftp/pub/corpora/tree-tagger1.pdf, 1994; www.ifi.unizh.ch/stff/siclemat/man/SchillerTeufel99STTS.pdf.

9. Schach, Stephen R.: An introduction to object-oriented analysis and design with UML and the unified process, McGraw Hill, Boston, Mass., 2004.
10. Schiller, A.; Teufel, S.; Stöckert Ch.; Thilen Ch.: Guidelines für das Tagging deutscher Textcorpora mit STTS.
11. Tufis, Dan; Mason, Oliver.: Tagging Romanian Texts: a Case Study for QTAG, a Language Independent Probabilistic Tagger. Proceedings of the First International Conference on Language Resources & Evaluation (LREC), 1998, pp. 589–596.

Appendix 1

STTS (Stuttgart-Tübingen Tagset) vs. NIBA-Tagset and Feature-System (only some example-verbtags are compared):

STTS	Gloss	Example	NIBA tagset		Attribute values
VVFIN	finites Verb, voll	[ich] lese	tVag/2	Transitive	[ps1], [sg], [ind], [pres]
VVINF	Infinitiv, voll	gehen	iV	Intransitive	[inf], [stat1]
VVINF	Infinitiv, voll	ankommen	eV	Ergative	[inf], [stat1]
VVINF	Infinitiv, voll	trinken	tVag/2	Transitive	[inf], [stat1]
VVIZU	Infinitiv mit *zu*, voll	auszuatmen	iV	Intransitive	[inf], [stat2]
VVIZU	Infinitiv mit *zu*, voll	anzukommen	eV	Ergative	[inf], [stat2]
VVIZU	Infinitiv mit *zu*, voll	loszulassen	tVag/2	Transitive	[inf], [stat2]
VVPP	Partizip Perfekt, voll	gegangen, gelesen			[stat3]
VAFIN	finites Verb, aux	[du] bist, [wir] werden	AUX		
VAIMP	Imperativ, aux	sei [ruhig !]	Vcop		[imp]
VAINF	Infinitiv, aux	werden, sein	Vcop		[inf], [stat1]
VAPP	Partizip Perfekt, aux	gewesen			[inf], [stat3]
VMFIN	finites Verb, modal	dürfen	AUX		[mod], [ps1], [pl]
VMINF	Infinitiv, modal	wollen	AUX		[mod], [inf]
VMPP	Partizip Perfekt, modal	gekonnt, [er hat gehen] können			

Analogical Modeling with Bias — allowing Feedback and Centering

Christer Johansson and Lars G. Johnsen

Dept. of Linguistics and Literature
University of Bergen
N-5007 Bergen, Norway
{christer.johansson, lars.johnsen}@lili.uib.no

Abstract. We show a computationally efficient approximation (cf. [1]) of a full analogy model [2, 3], implemented in a computer program, and tested on the CoNLL2000 chunk tagging task [4], putting clause boundaries around mainly np and vp phrases. Our implementation showed to be competitive with other memory based learners. It deviates only slightly from the theoretical model. First, it implements a version of homogeneity check, which does not account fully for nondeterministic homogeneity. Second, it allows feedback of the last classification, and thirdly it allows centering on some central feature positions. Positions containing a) those parts-of-speech tags and b) those words that are to be given a chunk tag are given a weight which is given by how many match patterns that are equally or more general. A match on two centered features gives its patterns an extra weight given by the number of features. The results can be summarized as follows: a) using only lexical features performs below baseline. b) The implementation without anything extra, performs as the baseline for five parts-of-speech features, and centering improves the results. c) Feedback on its own does not improve results, while feedback + centering improves results more than just centering. Feedback on its own makes results deteriorate. The results exceed F=92, which is comparable with some of the best reported results for Memory Based Learning on the chunk tagging task.

1 Introduction

Analogical modeling (AM) is a (memory based) method to evaluate the analogical support for a classification [2, 3, 5]. Chandler [6] suggested AM as an alternative to both rule based and connectionist models of language processing and acquisition. AM defines a natural statistic, which can be implemented by comparisons of subsets of linguistic variables, without numerical calculations [5]. The natural statistic works as a selection mechanism, selecting those patterns in the database which most clearly points out a class for a novel pattern.

The original AM model compares all subsets of investigated variables. This may cause an exponential explosion in the number of comparisons, which has made it difficult to investigate large models with many variables (> 10) combined

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 57-68

with large databases. Johnsen and Johansson [1] gives an accurate approximation of AM, which considers all analogical support from the database. The essential simplification (ibid.) is that each exemplar in the database only contributes with its most specific match to the incoming pattern to be classified. This provides a basis for directly comparing Skousens model to other models of memory based learning (MBL). In MBL, an example E is classified as belonging to category C by computing the score of E by going through the whole database. Skousens model requires the computation of the full analogical set for E, which we can now show to be approximated with resources that are close to a linear search through the database. The Johnsen and Johansson [1] approximation reduces the time complexity of the full analogical algorithm, but it may also simplify the addition of mechanisms such as feedback of the last classification, and putting focus on central features. The results imply that memory based learning methods are related by their evaluations of the nearest match set, where typically MBL only selects nearest neighbors. The AM model always considers all members of the database.

We will demonstrate that the proposed approximation reaches a high level of performance on a popular tagging task [4], if the algorithm is extended by mechanisms to focus on relevant features, as well as a mechanism of feedback of the latest classification. Without these additional mechanism, analogical modeling gives fairly low performance on this type of larger scale tasks that involve ambiguity and selecting a best alternative.

The implementation deviates slightly from the discussed, theoretical model. First, it implements a sloppy version of homogeneity check, which does not account fully for nondeterministic homogeneity. Second, it allows feedback of the last classification, and thirdly it allows centering on some central feature positions. The positions containing a) the parts-of-speech tags and b) the words that are to be given a chunk tag are given a high weight, and their immediate left and right context are given a lower weight. The weights are multiplied together for every matching feature position.

The results can be summarized as follows: a) using only lexical features performs below baseline. b) The implementation without anything extra, performs as the baseline for five parts-of-speech features, and centering improves the results. c) Feedback on its own does not improve results, while feedback + centering improves results more than just centering. Feedback only makes results deteriorate. The results reach F=92, which is comparable with the best reported results for Memory Based Learning on the task.

We can show a computationally efficient approximation of a full analogy model, implemented in a computer program, and tested on the CoNLL2000 chunk tagging task. This showed to be competitive with other memory based learners.

An empirical confirmation of the computational complexity showed a very slow increase with an increased number of features, although processing times increased with more demands on memory, an effect which is likely due to limits on internal memory.

We are presently not using feature weighting, such as information gain, which typically works on the level of individual feature values. Future research involves working on a method for automatically finding the relevant variables, and finding optimal weights (focus) for these variables.

2 Background on AM

We will not go into details of analogical modeling, beyond what is necessary for comparing it with memory based learning. Johnsen & Johansson [1] showed that the outcome in AM can be determined by summing up scores for each match pattern, where we only have to match the input once with all the examples in the database.

Examples in the database and each new input are expressed by a vector of feature values, similar to standard MBL. The operation of AM depends on matches. Each feature value may either match, between an example and the new input, or not. This creates a match vector where matches are encoded with a 1 and non-matches with 0, for example $< 0, 1, 0, 1, 1 >$ for five features.

We may imagine these vectors as a pointer to a box where we collect all the corresponding outcomes in the database. After we have gone through the database, we can look in all the non-empty boxes (which typically is of a much lower number than the number of examples), and observe the distribution of the outcomes. We are interested in those boxes that contain only one outcome. We call these boxes *first stage homogeneous*. Boxes with more than one outcome are less important, and may be discarded if we find homogeneous boxes pointed to by a more specific context, i.e. a match vector with more matches. The remaining (non-empty) boxes need to be sorted according to how many matches the index pattern contains. A more general pattern (e.g. $< 0, 0, 1 >$ is either homogeneous for the same outcome as the more specific pattern that it dominates (e.g. $< 1, 0, 1 >$, $< 0, 1, 1 >$, or $< 1, 1, 1 >$), or it is indeed *heterogeneous* and should be discarded.

A score$(\theta(x))$ is summed up for the number of homogeneous elements it dominates. Each part in the summation corresponds to looking in one of the above mentioned "boxes" (x). Each score for each box has an associated constant c_x, which would give us the exact value for full analogical modeling, if it was known.

The scoring of the analogical set expressed in mathematical notation is:

$$\sum_{x \in \mathcal{M}} c_x score(\theta(x)) \tag{1}$$

where $\mathcal{M}$ is the match set, and x is a context in the match set.

The implication of the work in [1] is that the match set $\mathcal{M}$, which is simple to calculate, contains *all* the results necessary for computing the overall effect, without actually building the whole analogical structure. In order to accurately weigh each context we need to estimate how many extensions each homogeneous pattern has. Johnsen and Johansson [1] develops a maximum and minimum

bound for this, and also discusses the possibilities for using Monte Carlo methods for discovering a closer fit.

Let us start with a simple and hypothetical case where M has exactly two members x and y with a different outcome. Any supracontextual label shared between x and y will be heterogeneous. The number of these heterogeneous labels are *exactly the cardinality of the power set of the intersection between x and y*. To see this, consider an example

$$\tau = (c, a, t, e, g, o, r, y)$$

and let x and y be defined as (using supracontextual notation):

$$\begin{array}{l} x = (c, -, t, -, -, o, r, -) \\ \text{with unique } score(\theta(x)) = (3, 0) \approx 3\ r \\ y = (c, a, -, -, -, o, -, -) \\ \text{with } score(\theta(y)) = (0, 8) \approx 8\ e \end{array} \tag{2}$$

Their common and therefore heterogeneous supracontextual labels are

$$\left.\begin{array}{l} (c, -, -, -, -, o, -, -) \\ (c, -, -, -, -, -, -, -) \\ (-, -, -, -, -, o, -, -) \\ (-, -, -, -, -, -, -, -) \end{array}\right\} \tag{3}$$

The total number of elements that dominate x is sixteen; the homogeneous labels out of these sixteen are those that dominate x and no other element with a different outcome; in this case y. The labels x shares with y are the four labels in (3), and x has 16-4=12 homogeneous labels above it. How is that number reached using sets?

Viewed as sets, the elements x and y are represented as:

$x = \{c_1, t_3, o_6, r_7\}$ and $y = \{c_1, a_2, o_6\}$.

Their shared supracontexts are given by the power set of their common variables.

$$\begin{array}{l} x \cap y = \{c_1, t_3, o_6, r_7\} \cap \{c_1, a_2, o_6\} = \{c_1, o_6\} \\ \mathcal{P}(x \cap y) = \\ \quad \mathcal{P}(\{c_1, o_6\}) = \{\emptyset, \{c_1, o_6\}, \{o_6\}, \{c_1\}\} \end{array} \tag{4}$$

This set has four elements all in all, which all are equivalent to the labels in (3). The sets in (4) represent the heterogeneous supracontextual labels more general than either x or y and these are the only heterogeneous supracontexts in the lattice Λ of supracontextual labels, given the assumptions made above.

The power sets for x and y have 16 and 8 elements respectively, so the total number of homogeneous supracontextual labels more general than either x or y is the value for the coefficients c_x and c_y from (1) calculated as:

$$\left.\begin{array}{l} c_x = \|\mathcal{P}(x) - \mathcal{P}(x \cap y)\| = 16 - 4 = 12 \\ c_y = \|\mathcal{P}(y) - \mathcal{P}(x \cap y)\| = 8 - 4 = 4 \end{array}\right\} \tag{5}$$

Plugging these numbers into the formula (1) gives the score of the analogical set for this case:

$$\begin{aligned}&\sum_{x \in M} c_x score(\theta(x)) = \\ &= 12 score(\theta(x)) + 4 score(\theta(y)) \\ &= 12\,(3,0) + 4\,(0,8) \\ &= (36,0) + (0,32) \\ &= (36,32)\end{aligned} \tag{6}$$

In the general case however, the set $\mathcal{M}$ consists of more elements, complicating the computation somewhat. Each $x \in \mathcal{M}$ may share a number of supracontextual elements with other elements of $\mathcal{M}$ that have a different outcome. The situation may be as depicted in the following table, where columns are labelled by elements of $\mathcal{M}$ (in boldface) with their associated hypothetical outcomes (in italics).

Table 1. Accessing disagreement in $\mathcal{M} \times \mathcal{M}$

$\mathcal{M}$	$\mathbf{a}_r$	$\mathbf{g}_e$	$\mathbf{h}_r$	$\mathbf{d}_f$
$\mathbf{a}_r$		$\mathcal{P}_{(a \cap g)}$		$\mathcal{P}_{(a \cap d)}$
$\mathbf{g}_e$	$\mathcal{P}_{(g \cap a)}$		$\mathcal{P}_{(g \cap h)}$	
$\mathbf{h}_r$		$\mathcal{P}_{(h \cap g)}$		$\mathcal{P}_{(h \cap d)}$
$\mathbf{d}_f$	$\mathcal{P}_{(d \cap a)}$	$\mathcal{P}_{(d \cap g)}$	$\mathcal{P}_{(d \cap h)}$	

Each cell in Table 1 is associated with the power set $\mathcal{P}(x \cap y)$. This power set is only computed if the outcome of x is not equal to the outcome of y, and both outcomes are unique. The intersection is computed for all labels with a non-unique outcome, even for those with identical outcomes. If two elements are non-unique, any label that is a subset of both will match up with their respective and disjoint data sets (see propositions 1 and 2 in [1]), thereby *increasing* the number of disagreements, and consequently turning any such label into a heterogeneous label. Note that a and h have the same outcome in this table making their intersective cells empty.

Each non-empty cell corresponds to the simple case above. The complication stems from the fact that different cells may have non-empty intersections, i.e., it is possible that

$$\mathcal{P}(a \cap g) \cap \mathcal{P}(a \cap d) \neq \emptyset$$

Arithmetic difference of the cardinality of the cells may be way off the mark, due to the possibility that supracontexts may be subtracted more than once.

Something more sophisticated is needed to compute the desired coefficients c_x. A couple of approximations are given in the following.

The approximations are gotten at by first collecting all subcontexts different from $a$in a set $\delta(a)$:

$$\delta(a) = \{x \in \mathcal{M} | o(a) \neq o(x)\}$$

This equation represents the column labels for the row for a. The total number of homogeneous supracontexts ($=c_a$) more general than a is the cardinality of the set difference

$$\mathcal{P}(a) - \bigcup_{x \in \delta(a)} \mathcal{P}(a \cap x) \tag{7}$$

The second term in (7) corresponds to the value of the function H in [1], and is the union of the power sets in the row for a. It represents the collection of supracontextual labels more general than a, which also are shared with another subcontext, thus making all of them heterogeneous. The first term, $\Pi(a)$, is the set of all supracontextual labels more general than a. Therefore, the difference between these two sets is equal to the collection of homogeneous supracontextual labels more general than a. However, it is not the content of these sets that concerns us here; the goal is to find the cardinality of this difference.

The cardinality of $\mathcal{P}(a)$ is given the normal way as

$$\|\mathcal{P}(a)\| = 2^{\|a\|}$$

but how are the behemoth union to be computed? This raises the question of computing the union of power sets:

$$\bigcup_{x \in \delta(a)} \mathcal{P}(a \cap x) \tag{8}$$

The exponential order of the analogical algorithm stems from the computational complexity of this set. The union is bounded both from below and above. A lower bound is:

$$\mathcal{P}(max(\{a \cap x | x \in \delta(a)\})$$

and a higher bound is:

$$\mathcal{P}(\bigcup_{x \in \delta(a)} a \cap x)$$

Both these bounds are fairly simple to calculate. In the implementation (written in C), we have chosen a weighted average between the lower bound and the higher bound as a good approximation. We found that values that are weighted in favor of the higher bound gave better performance. This is not equivalent to say that the true AM values are closer to the higher bound.

3 Results from the Implementation

We have evaluated the performance of our implementation using the chunk identification task from CoNLL–2000 [4]. The best performance was obtained by a Support Vector Machine [7, F=93.48]. This implementation has the disadvantage that it is computationally very intensive, and it might not be applicable to much larger data sets. Their results have later improved even more for the same task. The standard for memory based learning on this task is an F value of 91.54 [8], a value which can be improved upon slightly, as is shown by using a system combining several different memory-based learners [9, F=92.5]. Johansson [10] submitted an NGRAM model which used only 5 parts-of-speech tags , centered around the item to be classified. That model (ibid.) used a backdown strategy to select the largest context attested in the training data, and gave the most frequent class of that context. It used a maximum of 4 look-ups from a table, and is most likely the fastest submitted model. The table could be created by sorting the database. The advantage being that it could handle very large databases (as long as they could be sorted in reasonable time). The model gives a minimum baseline for what a modest $NGRAM$-model could achieve (F=87.23) on the chunking task.

3.1 Deviations from AM

The implementation deviates slightly from the discussed, theoretical model. First, it implements a version of homogeneity check, which does not account for non-deterministic homogeneity [2, 3, 5]. We tried a more extensive homogeneity check in an earlier version, but the results actually deteriorated. Second, it allows feedback of the last classification, and thirdly it allows centering on some central feature positions. The positions containing a) the parts-of-speech tags and b) the words that are to be given a chunk tag are given a weight given by how many more general patterns exist.

Giving Proper Weight on the Focussed Items The number of patterns with a lower or an equal number of hits is given by:

$$a) \sum_{k=0}^{hits} \binom{n}{k} \quad b) \; n \sum_{k=0}^{hits} \binom{n}{k} \quad c) \sum_{k=0}^{hits} \binom{n}{k} / n; \tag{9}$$

The term *hits* refers to the number of matching features in the match pattern we are considering. Formula a) in 9 refers to the case where we have found one centered feature. Formula b) refers to when both centered features have been found, and finally formula c) refers to when none of the centered features have been found. In the case that none of the features have been found the effect is spread evenly over the available variables (feature positions). If one matching centered feature is found the effect is concentrated to one, and if both centered features match we might have choosen the second centered feature in $n-1$ ways. We have added 1, in part to make the formulas more uniform.

3.2 Performance

From Table 2 we can see that using only lexical features (i.e. 5 lex and 6 lex), performs below baseline (F=87.23). The implementation without anything extra (F=87.65), performs only slightly better than the base line for five parts-of-speech features, and centering improves that to 88.50. Feedback on its own has a small effect (87.39; 6 pos, 82.95 6 lex), while feedback + centering (F=89.03: 6 pos, 87.69; 6 lex) improves results compared to just centering. Feedback does introduce some mistakes from the mechanism, and centering allows some of those to be corrected.

Table 2. Results: F-scores. # features, Feedback + Centering, Centering only, Feedback only, nothing extra. Results within () are results without the binomial weighting from eqn. 9c

#	F+C	C	F	0
5 lex		85.95		82.50 (80.40)
5 pos		88.50		87.65 (87.13)
6 lex	87.69		82.95 (83.17)	
6 pos	89.03		87.39 (87.02)	
10		91.45		89.69 (89.48)
11	92.23		89.58 (89.41)	

The model with only centering using both lexical and parts-of-speech features approaches MBL results, and performs slightly better with feedback and centering (F=92.23, see Table 3), although not as good as the SVM-implementation [7], nor the system combining several memory based learners [9].

Table 3. Detailed results for each category. (11 F+C)

selected	precision	recall	$F_{\beta=1}$
ADJP	72.91%	67.58%	70.14
ADVP	77.87%	78.41%	78.14
CONJP	36.84%	77.78%	50.00
INTJ	100.00%	50.00%	66.67
NP	92.15%	93.29%	92.72
PP	95.91%	97.38%	96.64
PRT	71.72%	66.98%	69.27
SBAR	87.82%	82.24%	84.94
VP	92.21%	92.74%	92.48
accuracy	precision	recall	FB1
95.13%	91.86%	92.60%	92.23

We have also tried out an idea of providing a model for novel items by constructing instances where low frequency words were replaced by a marker common to unseen words in the test set. This idea resulted in the same, or slightly worse results. This indicates that the algorithm does not get any new information from these added constructions, i.e. the information was already available, and it showed to be fairly hard to alter the classification by adding items to the database.

3.3 Computational Performance and Expected Complexity

The time complexity of the abstract algorithm for the worst case was asserted to be $O(log(N)N$ [1]. The current implementation has a nested loop over the match set, which in the worst case may grow to be as large as the database. This would make the algorithm $O(N^2)$ in the worst case. We do not expect that to happen in the average case. What was the performance on the *CoNLL* task?

The tests were made using a 867MHz PowerPC G4, with 1 MB L3 cache and 256 MB SDRAM memory using Mac OS X version 10.3.9.

When the number of variables changed, the number of unique patterns varied. The time to process all test patterns were therefor divided by the number of unique database items and reported as how many milliseconds per database item the processing took.

The results are shown in Table 4. This shows an almost linear increase with the number of variables, which has to do with a) that more comparisons are made because there are more variables (and features values) to compare, and b) that the match set $\mathcal{M}$ grows slightly faster when there are more variables. A deviation from the linear marks this increase in match set growth.

Table 4. Processing time needed to solve the full task, per item in the database.

#	*D*	ms
5 lex	213532	17.44
5 pos	92392	17.84
6 lex	213562	19.14
6 pos	92392	19.80
10	213562	26.07
11	213591	28.68

When the size of the database is accounted for, the main contribution to complexity is proportional to the square of the size of the match set times the number of variables. That time expenditure does not grow faster indicates that the match set does not grow very fast for this task. The values in Table 4 are approximated by the formula: $time = 0.05 * f^2 + f + 11.5$, with $R^2 > 0.98$; f = number of variables, and time is in ms per database item, for processing all 49393 instances in the test set.

4 Future Research

The relevant features for focus can be found automatically in many cases, by looking at how the variable in general correlates with the outcome. This may expand the model to consider more than two focussed variables.

We are presently not using feature weighting, such as information gain, which typically works on the level of individual feature values. This might give some room for future improvement.

5 Conclusion

We have shown an outline of a theoretical reconstruction of Skousen's Analogical Modeling of Language [2, 3, 5], this is described in more detail elsewhere [1, 11]. This reconstruction led to a more efficient approximation of full analogy modeling, and the results were implemented in a computer program, and tested on the *CoNLL* – 2000 chunk tagging task. Our implementation showed to be competitive with other memory based learners, after accounting for the specificity of the match patterns and how many patterns were more general than the pattern under consideration.

Admittedly, non-naive implementations of a nearest neighbor model, such as TiMBL [12], are already doing well for large data sets, which make it hard to compete on combined accuracy and processing time. The main contribution of this model is that it implements a memory based model without the need to specify how many nearest neighbors (or neighbor distances) to consider. In the spirit of AM there are no parameters to set, and no calculation of gain for knowing some individual item. We have added the possibility to center on particular variables, to separate them from context variables. We have also provided the possibility to use feedback of the last categorization. As there are some approximations involved, we have made it possible to alter the weights to make it possible to optimize the results. The highest performance reached so far has been F=92.25 (compared to the reported F=92.23 for the standard settings).

The implementation has reached its level of performance without calculating the information gain of knowing some individual feature values. In this, the implementation follows the philosophy of parameterless analogical modeling. It should be interesting for linguistic models that a model based on selection can reach fairly high performance without calculating any statistics based on the individual items.

Acknowledgement Support by a grant from the Norwegian Research Council under the KUNSTI programme (project BREDT) is kindly acknowledged. The computer program will be made available for download from http://bredt.uib.no with some example data.

References

1. Johnsen, L., Johansson, C.: Efficient modeling of analogy. In Gelbukh, A., ed.: Proceedings of the 6th Conference on Intelligent Text Processing and Computational Linguistics. Volume 3406 of Lecture Notes in Computer Science. Springer Verlag, Berlin, Germany (2005) 682–691
2. Skousen, R.: Analogical Modeling of Language. Kluwer Academic, Dordrecht, the Netherlands (1989)
3. Skousen, R.: Analogy and Structure. Kluwer Academic, Dordrecht, the Netherlands (1992)
4. Tjong Kim Sang, E., Buchholz, S.: Introduction to the CoNLL-2000 shared task: Chunking. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 127–132
5. Skousen, R., Lonsdale, D., Parkinson, D.: Analogical Modeling: An exemplar-based approach to language. Volume 10 of Human Cognitive Processing. John Benjamins, Amsterdam, the Netherlands (2002)
6. Chandler, S.: Are rules and modules really necessary for explaining language? Journal of Psycholinguistic Research **22** (1993) 593–606
7. Kudoh, T., Matsumoto, Y.: Use of support vector learning for chunk identification. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 142–144
8. Veenstra, J., Bosch, A.v.d.: Single-classifier memory-based phrase chunking. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 157–159
9. Tjong Kim Sang, E.: Text chunking by system combination. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 151–153
10. Johansson, C.: A context sensitive maximum likelihood approach to chunking. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 136–138
11. Johnsen, L.: Commentary on exegesis of Johnsen and Johansson, 2005, (ms.) (2005)
12. Daelemans, W., Zavrel, J., van der Sloot, K., van den Bosch, A.: TiMBL: Tilburg Memory Based Learner, version 5.1. Reference guide. ILK Technical report Series 04-02., Tilburg, the Netherlands (2004)

Word Sense Disambiguation and Semantics

Word Sense Disambiguation with Basic-Level Categories

Steve Legrand

Department of Computer Science, University of Jyväskylä, Finland
stelegra@cc.jyu.fi

Abstract. Research in basic-level categories has provided insights that can be beneficially used in word-sense disambiguation. Human beings favor certain categories over others and this is reflected in their use of language. Parts and functions categories are especially useful in providing contextual clues. The disambiguation effort in this article concentrates on the two main senses of the word "palm." The results are encouraging and indicate that basic-level categories will have a role to play in computational linguistics.

1 Introduction

If a word has only one sense, a non-native speaker can confirm its meaning by a quick look at a dictionary. Most of the words do have, however, more than one sense, and both the native and the non-native speaker need to use the word context in order to find its correct sense. For example, when we look at the sentence,

There was a large blister on the heel of his right palm.

it is obvious to us that the word *palm* refers to a body part rather than to a tree or a handheld computer. The words *blister*, *heel*, *his*, and *right* when combined in a certain way point us towards the correct meaning.

Most of the automated disambiguation techniques, one way or another, are context-based, making use not only of the words themselves, but also of the part-of-speech information, word order, document genre and so on. Generally, we can say that these techniques are justified by our observations that certain words do co-occur quite regularly with each other within certain contexts. This notion has been used somewhat heuristically in automated word sense disambiguation, and often there is no reference to any cognitive disambiguation mechanism that could have been involved. Nevertheless, it is not disputed that context plays a very important part in the word sense disambiguation by our cognitive faculties.

The question arises: what is this human disambiguation mechanism like if it exists, and would it be possible to mimic and exploit it in automated word sense disambiguation? Is it rooted in our biology, and consequently reflected in our cognitive abilities, including our ability to categorize? The classical view of categories is often interpreted as meaning that things belong to the same category only if they have certain properties in common. It might seem that car parts such as a wheel and an engine do not share any properties, therefore should one assume that they cannot belong to the

© *A. Gelbukh (Editor)*
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 71-82

same category? On the closer inspection one can, however, discover, that they have at least one common property, and that is that they are parts of a car. So partonomy can create categories of things that apparently do not have much to do with each other. In fact, we can divide and subdivide our universe in so many different ways that what we know as classical categorization may prove inadequate for many tasks, including word sense disambiguation. Using the Family Resemblance Theory [30], basic-level categories [3] and experiments demonstrating these theories [21], Lakoff [13] challenges the classical view of categorization, proposing to correct it with a move to idealized cognitive models (ICMs) based, to a large extent, on prototype-level categories.

Here we will demonstrate that the type of categorization, "a human view of the world", that Rosch and Lakoff favour, may indeed be reflected in the language that we use to describe things, and, therefore, can benefit word sense disambiguation. The work is still at its preliminary stages, and the purpose of this paper is merely to explain the theoretical basis behind it and illustrate it with a simple example.

In what follows, we will go briefly through the concepts of basic-level categories (Section 2), idealized cognitive models (Section 3) and ontology structures (Section 4) before explaining how to use refined InfoMap [11] results for creating an ontology that may be more suitable for word sense disambiguation (Section 5). Finally, to exemplify our suggested approach, the two major senses of the word *palm* are disambiguated (Section 6). A more extensive study is underway, the results of which will be published shortly.

2 Basic-Level Categories

Roger Brown [3] explained his notion of a "first level" as a kind of category which allows children to learn object categories and name them, but which, as a category, falls somewhere between the most general and the most specific level. Later Rosch [21, 22] designed a series of experiments in which she demonstrated that the basic-level categories, as she started calling them, were somewhat inconsistent with the classical theory of categories, and she explained their specific properties:

From the point of view of human cognition, the categories seem to be divided roughly into three kinds: superordinate (*furniture*), basic-level (*chair, table, lamp*), and subordinate (*kitchen chair, living-room chair / kitchen table, night table / floor lamp, desk lamp*). The basic-level objects have most of the attributes that are common to all members of the category and they share the least number of attributes with other, contrasting categories. Category membership is also influenced by family resemblances [30] to prototypical members. Archambault et al [1] in their brief review of literature selected the following (most of it also investigated by Rosch or based on her research) as the most important issues to note about basic-level categories:

- Categories at the basic-level are verified fastest.
- Objects are named faster at the basic than at the subordinate level.
- Objects are preferentially named with their basic-level names.
- Basic-level names are learned before subordinate names.
- Basic-level names tend to be shorter.

Tversky and Hemenway [27, 26] propose that parts may play a major role in the recognition of basic-level objects, which may have something to do with the so-called Gestalt peception [12] related to part-whole configuration. In their proposal there is a strong suggestion that our basic-level object perception may well be based around this part-whole division. Parts, in turn, are related to functions, shape and interactions of these basic-level objects. This gives rise to an interesting question: is the categorization around parts reflected in the language we use?

3 Idealized Cognitive Models

Lakoff [13] believes that linguistics categories have the same character as other conceptual categories: they show prototype effects and can be demonstrated to have basic-level categories. But he makes it clear that neither he nor Rosch advocate the view that basic-level categories would explain any structural or procedural properties of cognition. Rather, they both regard basic-level categories as a mere surface phenomena related to cognition, and assume that below that surface there may be some other more interesting structures and processes to be found.

Lakoff's main thesis is that our knowledge is organized by means of structures to which he refers to as idealized cognitive models, or ICMs, and that category structures and prototype effects are their by-products. Each ICM is seen as a structured whole, a gestalt, with four structuring principles employed:

- propositional structure (Fillmore's [7] frames)
- image-schematic structure (Langacker's [14] cognitive grammar)
- metaphoric mappings
- metonymic mappings

These ICMs would then structure the mental space as described by Fauconnier [6]. As examples of ICMs, among others, Lakoff refers to a Balinese calendar system with three different "*week*" structures superimposed [9], the category defined by the English word *bachelor* [7] and other examples.

The importance of Lakoff's ICMs to this research is in that he shows how, by extending the basic-level categories to the linguistic domain, we can end up with novel categorical structures, which may have not been considered at all in the creation of ontologies that are widely used today. This, in turn, may be one of the reasons why these conventional ontologies may prove inadequate for linguistics tasks such as word sense disambiguation.

4 Example Ontology — WordNet

WordNet 2 defines itself as "a machine-readable lexical database organized by meanings". It organizes English nouns, verbs and adverbs into synonym sets representing lexical concepts [8]. The sets are linked by relations such as hypernym, meronym, synonym and antonym. WordNet has been critizised for not providing a useful organ-

isational principle for information retrieval, reasoning, or knowledge management, being based on linguistic rather than encyclopaedic coherence [2]. Concepts likely to occur together in a domain are often found widely separated from each other in the conceptual hierarchy [24].

However, the linguistic principles employed in WordNet's construction have made it a useful tool for word sense disambiguation. WordNet has been used with many different WSD techniques, the resulting disambiguation accuracies ranging from 57% to 92% [4, 16, 19, 20]. To make it even more useful for WSD, some important cognitive principles might need to be explicitly added to its organization. These could be implemented through pointers as ontological relations.

In fact, the authors of WordNet had this in mind when starting to construct it. As an example, Miller pointed out that the word *canary* should be associated with at least three types of distinguishing features: (1) attributes (small, yellow and other adjectives), (2) parts (beak, wings and other nouns), and (3) functions (sing, fly, and other verbs). The addition of the distinguishing features important to basic-level categories was contemplated, but was not implemented explicitly except for the pointers to the parts [18]. Instead, glosses were added which contain some of these features. Many WSD implementations have used these glosses since for sense disambiguation.

In this research, feature sets incorporating these distinguishing features and also other associations and collocations are used. Most of these are not explicitly expressed in WordNet, and here we try roughly to gauge their relative importance to WSD.

5 Use of InfoMap as the First-Stage Disambiguator

To disambiguate with the help of context one needs a set of words that co-occur, more often than would be the case by chance, with the word to be disambiguated. One way to do this would be to collect co-occurrence statistics with whatever software were available for the purpose, but the drawback of this method is that the statistics do not discriminate between the senses. A better way is to use an application that is based on co-occurrences but which, nevertheless, can be made to discriminate, to some extent, between the senses when a judicious selection of the search terms is performed. One such application is InfoMap (http://infomap.stanford.edu), which is freely available from the Stanford University site and is explained in detail in [25, 29].

The principle behind InfoMap was developed by Schütze [23] and implemented and modified principally by the InfoMap team at the Stanford University. The distribution of word co-occurrences between a word and sets of content-bearing words creates a profile of the words usage in a context, and thus a profile of the word meaning itself. A similarity between two words can be calculated by comparing the profiles of the words in question. It is possible to return related documents whose profiles are close to each other even though they may not include the query words themselves. The meaning can be narrowed down by the selection of search words and can thus be used to disambiguate the key search term to some extent at least.

To get a set of word clusters related to the word palm in the sense of a hand (Table 1) we can simply enter the words *palm* and *hand* together as search terms, together with any negative keywords. The web interface allows us to retrieve up to 200

Table 1. Results of an InfoMap-query using *palm* and *hand* as keywords and *tree* as a negative keyword. 10 clusters were specified.

Prototypical Example	Cluster Members
hand	hand wrist elbow finger thumb forearm grasped glove holding squeezed grasping firmly coin ear torch cigarette lever grasp isambard pencil verbal button squeeze candle undone propped superiority mister tapped arm's
palm	palm tapping held knuckles squeezing aloft wrapped knotted cradle shield caf woven loom restraining cloth smacked clips begging salute raffle necklace delights twists cane embroidered
fingers	fingers cheek cupped clutched stroked grip touched stroking brushed gripped lightly kissed gently tenderly fingertips delicately hold flinched knife stretched touch rubbing rested touching blade lifted pins dagger limp slid knelt shake caress razor pressed gasping tip rope raised brush
shoulder	arm shoulder outstretched sleeve fist clasped clutching clamped thigh waved sword knee patted foot hip gripping rein gesture hips trouser knob leg reins swinging breast smoothing bend needle forward
pocket	pocket put picked wallet handed bag tore crumpled briefcase pen drawer pad cardboard paw pockets parchment suitcases handbag putting lend packet
left	left fork hemisphere edge side scars stile pictured
grabbed	grabbed gun pistol snatched fumbled wrenched grab
lips	tightly rubbed chin trembling clenched mouth handkerchief kiss boy's lips breath gasped brow twitched
jar	saucer basket teapot bottle plate crumbs biscuit jar tray
shoulders	forehead shaking bent resting waist arms jerked tugging tilting rolled curled chest palms slapped knees wrists shoulders

words associated with the key search terms and divide them into 1–20 categories as desired. A prototypical example is given for each category. Other search strategies could also be used for the same end including contrasting pairs.

6 Basic-Level Categories and ICM's in WSD

The idea behind using InfoMap is to get a set of terms associated with the word to be disambiguated and occurring together in the same context. InfoMap is based on co-occurrence information and word vector relations and, therefore, seems suitable for the purpose. The public web interface for the application at the Stanford University site was included within the Java-based disambiguating application created for the purpose. The mode of the operation was, shortly, as follows.

The parameters posted to the site were the search term palm + other keywords., (hand), negative keywords (tree), corpus (British National Corpus), command (associate), and parameters specifying clustered results with 200 words divided in 10 clusters. The request to the site was sent separately for both of the major senses of the word to be disambiguated and the results received were combined to form a Disam-

biguation Feature Cluster set consisting of 20 clusters, the first 10 for the first major sense of the word to be disambiguated (keywords: palm hand, neg. keyword: tree) the second 10 for the second major sense (keywords: palm tree, neg. keyword: hand). The returned information (Figure 1 showing half of the set) was then converted into an XML-format and indexed into a file database using Java Digester Libraries [5]. The context to be disambiguated was indexed to another data base using Digester and Porter Stemming Algorithm. In the process of disambiguation, the context sentences were iterated through and matched against the Disambiguation Feature Cluster set: each time a word in the context sentence matched the clusters 1–10 the first sense increased its score, and when 11–20 were matched the second sense increased its score. The maximum of these scores indicated the word sense. The matches were indicated either as correct, undecidable (no matches), even, or wrong. For the query matching, Java Lucene [10, 17] libraries were used.

First we tested our application with Mihalcea's sense tagged data for six words with two-way ambiguities, previously used in word sense disambiguation research and extracted from BNC [28]. We simply took her Meanings-labels as positive and negative keywords to create the feature-sets with the help of InfoMap and then used these feature-sets to disambiguate her examples. The results are shown in Table 2. As expected, the results were variable, ranging from 47.3 to 82.2 % in accuracy, indicating that the selection of the keywords is significant. Changing *tank*'s "vehicle"-keyword to "military," for example, increased the disambiguation accuracy to 64.7%. Increasing the number of the keywords also had a significant effect on the result.

Table 2. Disambiguation accuracies reached using the TWA dataset's Meanings-labels as keywords

Word	Meanings	Examples	Correct
bass	fish/music	107	82.2 %
crane	bird/machine	95	68.4 %
motion	movement/legal	201	49.8 %
palm	hand/tree	201	72.0 %
plant	living/factory	188	77.1 %
tank	container/vehicle	201	47.3 %

However, our purpose was not to find out the maximum disambiguating power of InfoMap , but to use it as a tool to help in our own experiments. We merely needed a rough set of context words to modify using basic-level category information to see how that information affected the disambiguation accuracy.

For our example, the word *palm* was selected, because it had an adequate number of hand-tagged contexts (201) and the disambiguation accuracy (75.1%) achieved was judged sufficient, but not too, high for our purpose. Moreover, we could extract an adequate number of contexts (1000) with the word *palm* from the BNC against which to test this set. As both sets come from the BNC, they may partially overlap, but, as said, the purpose of the experiment was to test the effect of the basic-level category words on the overall disambiguation against normal context words. More extensive n-way tests will follow based on this experiment. After pruning out some minor senses, 193 contexts remained. The remaining TWA contexts were processed using the un-

Table 3. Results for the two major senses (part-of-hand, tree) of the word *palm* in 193 contexts when disambiguated with the unmodified disambiguation feature cluster set (UDFC).

Wide context (paragraph)			Narrow context (sentence)		
correct:	139	72.0 %	correct:	130	67.4 %
undecidable:	0	0.0 %	undecidable:	0	0.0 %
equal:	24	12.4 %	equal:	22	11.4 %
wrong:	30	15.5 %	wrong:	41	21.2 %
Total:	193	~100.0 %	Total:	193	~100.0 %

Table 4. Results for the two major senses (part-of-hand, tree) of the word *palm* in 193 contexts when disambiguated with the MDFC set.

Wide context (paragraph)			Narrow context (sentence)		
correct:	193	100.0 %	correct:	193	100.0 %
undecidable:	0	0.0 %	undecidable:	0	0.0 %
equal:	0	0.0 %	equal:	0	0.0 %
wrong:	0	0.0 %	wrong:	0	0.0 %
Total:	193	~100.0 %	Total:	193	~100.0 %

modified InfoMap feature set for disambiguation. The results were as shown in Table 3.

Even when disambiguating with the unmodified InfoMap results, the disambiguation achieved is significantly better than what could be expected by chance. Our purpose was to modify the feature set to see what the actual words were that played role in disambiguation and what was their number, in order to be able to roughly categorize the words participating in disambiguation. For this reason the words that had not played any part in disambiguation, were pruned from the Disambiguation Feature Cluster Set. Some words that were judged as missing were added, and to get a 100% disambiguation result for the TWA contexts (Table 4) further 5 collocations ({"his","palm"}, {"read","palm"}, {"her","palm"}, {"my","palm"}, {"palm","tree"}) were added. The number of the words in the modified and unmodified set remained roughly the same. We call the original, unmodified set the Unmodified Disambiguation Feature Clusters (UDFC) set and the modified one the Modified Disambiguation Feature Clusters (MDFC) set. In the MDFC the feature categories were rearranged to create additionally a feature set for a) Parts, b) Objects Affected, and c) Functions in order to roughly isolate the features that might be related to basic-level information.

1000 contexts containing the word *palm* were then extracted from the BNC out of which 749 contained either of the major senses (part-of-hand, tree) and these were then selected for disambiguation. First these contexts were disambiguated with the help of the UDFC set (Table 5) and then with the help of the MDFC set (Table 6).

As these results show, the disambiguation accuracy for the MDFC was considerably higher than for the UDFC. The accuracy of UDFC increased when the number of contexts was increased, whereas the accuracy declined for MDFC. This probably was due to the fact that MDFC was optimized for TWA contexts whereas UDFC was not, i.e., some of the pruned words might have proved useful in new contexts, etc.

Table 5. Results for the two major senses (part-of-hand, tree) of the word *palm* in 749 contexts when disambiguated with the UDFC set.

Wide context (paragraph)			Narrow context (sentence)		
correct:	596	79.6 %	correct:	550	73.4 %
undecidable:	2	0.3 %	undecidable:	0	0.0 %
equal:	71	9.5 %	equal:	76	10.1 %
wrong:	80	10.7 %	wrong:	123	16.4 %
Total:	749	~100.0 %	Total:	749	~100.0 %

Table 6. Results for the two major senses (part-of-hand, tree) of the word *palm* in 749 contexts when disambiguated with MDFC set.

Wide context (paragraph)			Narrow context (sentence)		
correct:	702	93.7 %	correct:	692	92.4 %
undecidable:	11	1.5 %	undecidable:	32	4.3 %
equal:	13	1.7 %	equal:	12	1.6 %
wrong:	23	3.1 %	wrong:	13	1.7 %
Total:	749	~100.0 %	Total:	749	~100.0 %

Then a very rough estimation was made of the contribution that the feature-sets (Parts, Functions) linked to basic-level information (hypernyms, parts, functions) made towards the overall disambiguation. For this the 193 pruned contexts from TWA were used. First, the parts and functions clusters from the MDFC were removed and the remaining clusters only were used for disambiguation. As the word *palm*'s salience varied within the context, being sometimes in the foreground sometimes the background, it was decided to conflate the part information between adjacent levels: all tree parts were considered together and all body parts were considered together. Similarly, all tree function words were considered together, and all body function words were considered together.

The disambiguation accuracy exceeded the 50/50 (Table 7) with significant results, but a lot of scope was left for improvement, which shows that the inclusion of parts and functions in the clusters used in MDFC is essential for accuracy. This is shown even clearer when we include only the parts and functions clusters and remove others from MDFC (Table 8).

7 Discussion

Although, as the very first experiment with the unmodified set returned by InfoMap shows, the disambiguating word set needs to be modified for more accurate functioning, the size of the set (200 words for each sense) seems adequate. Our preliminary experiments with other disambiguous words have shown that an ontology relating words through structures including the novel categories would ideally suit for word sense disambiguation. We have previously successfully used WordNet for disambiguating words based on an artificial taxonomy (animals) [15] and expect that by aug-

Table 7. Results for the two major senses (part-of-hand, tree) of the word *palm* in 749 contexts when disambiguated with the MDFC set. Two MDFC clusters, parts, and functions, are not used in this MDFC.

Parts and functions clusters not included					
Wide context			Narrow context		
correct:	121	62.7 %	correct:	102	52.8 %
undecidable:	49	25.4 %	undecidable:	84	43.5 %
equal:	13	6.7 %	equal:	1	0.5 %
wrong:	10	5.2 %	wrong:	6	3.1 %
Total:	193	~100.0 %	Total:	193	~100.0 %

Table 8. Results for the two major senses (part-of-hand, tree) of the word *palm* in 749 contexts when disambiguated with the MDFC. Only parts, and functions clusters are used in this MDFC.

With parts and functions clusters only					
Wide context			Narrow context		
correct:	149	77.2 %	correct:	144	74.6 %
undecidable:	2	15.0 %	undecidable:	40	20.7 %
equal:	9	4.7 %	equal:	6	3.1 %
wrong:	6	3.1 %	wrong:	3	1.6 %
Total:	193	~100.0 %	Total:	193	~100.0 %

menting the relations within WordNet to include categorical relations that appear to have some relation with basic-level categories and idealized cognitive models we could make it more suitable for disambiguation purposes. However, there are many questions to be solved about the basic-level categories, ICMs and their relations to context before a more comprehensive system can be developed. For example, something perceived as basic level varies amongst individuals: for an expert *eucalypt* may appear as a basic-level object, whereas for many ordinary city-dwellers it is *tree* that is seen as the basic-level object. The salience of the word within the context, i.e., whether it is in the background or in the foreground, affects the gestalt experienced also. There may be hundreds of different types of ICMs judging by the variety of examples given by Lakoff and others. Some of this information is already coded in different ontologies, albeit referred to by different terms, such as thematic relations, partonymy etc. It is likely, as Rosch and Lakoff have pointed out that basic-level structures are a mere surface phenomena, and one needs to dig deeper to get to the gist of what happens in the cognition when dealing with categories, in order to allow us to build structures that can be used in disambiguation

Complex as it might seem considering the reservations above, the research is justified on the grounds that a human being can disambiguate linguistic context better than a machine, and unless we are able to come up with a superior algorithm or mimic this disambiguating behavior, we can never be sure whether the results that our machine translation and other applications come up with are correct. We need to communicate globally and rapidly and need to be able to do it without the fear of being misunderstood.

References

1. Archambault, A., Gosselin, F., Schyns, P. (2000). A natural bias for the basic level? Proceedings of the 22nd Annual Conference of the CognitiveScience Society, NJ: LEA, 585-590.
2. Brewster, C. (2002). Techniques for Automated Taxonomy Building: Towards Ontologies for Knowledge Management. In Proc. CLUK Research Colloquium, Leeds, UK.
3. Brown, R. (1958). How Shall a Thing be Called? Psychological Review 65:14-21.
4. Cañas, A.J., Valerio, A. Lalinde-Pulido, J., Carvalho, M., Arguedas, M. (2003). Using WordNet for Word Sense Disambiguation to Support Concept Map Construction Proceedings of SPIRE 2003 – 10th International Symposium on String Processing and Information Retrieval, October 2003, Manaus, Brazil.
5. Digester can be downloaded from: http://jakarta.apache.org/commons/digester/.
6. Fauconnier, G. (1985). Mental Spaces. Cambridge, Mass. MIT Press.
7. Fillmore, Charles J. (1982). Frame semantics. In Linguistics in the Morning Calm, Seoul, Hanshin Publishing Co., 111-137.
8. Fellbaum, C. ed. (1998). WordNet: An Electronic lexical database, MIT Press.
9. Geertz, C. (1973). The Interpretation of Cultures. New York. Basic Books.
10. Gospodnetic, O., Hatcher, E., Lucene in Action, (2004), Manning Publications, Geenwich, CT.
11. InfoMap information mapping project web interface at http:// infomap.stanford.edu/.
12. Koffka, K. (1922). Perception: An introduction to the Gestalt-theorie. Psychological Bulleting, 19. pp. 531-585.
13. Lakoff, G. (1987). Women, Fire, and Dangerous Things. The University of Chicago Press. Chicago and London.
14. Langacker, R. (1986). Foundations of Cognitive Grammar, vol 1. Stanford Univ. Press.
15. Legrand, S., Pulido JGR. (2004). A Hybrid Approach to Word Sense Disambiguation: Neural Clustering with Class Labeling. Knowledge Discovery and Ontologies (KDO-2004) workshop in15th European Conference on Machine Learning (ECML) and 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD) Pisa, Italy, September 24, 2004
16. Li X., Szpakowics, S., Matwin, S. (1995). A WordNet-based Algorithm for Word Sense Disambiguation. Proceedings of IJCAI-95. Montréal, Canada, 1995.
17. Lucene Java Libraries can be downloaded from: lucene.apache.org/java/docs/ index.html.
18. Miller, G.A., Beckwith, R., Fellbaum, C.D, Gross, D., Miller, K. (1993). Five Papers on WordNet. Technical report, Princeton University, Princeton, N.J.
19. Mihalcea, R., Moldovan, D. (2000). An Iterative Approach to Word Sense Disambiguation, Proc. of Flairs 2000, pp. 219-223, Orlando, FL.
20. Nastase, V., Szpakowics, S. (2001). Word Sense Disambiguation in Roget's Thesaurus Using WordNet, Proc. of the NAACL WordNet and Other Lexical Resources Workshop, Pittsburgh, June 2001.
21. Rosch, E., Mervis, C.B..Gray, W.D., Johnson, D.M., and Boyes-Braem, P. (1976). Basic objects in natural categories. Cognitive Psychology, 23, 457-482.
22. Rosch, E., Principles of Categorization. (1988). In Readings in Cognitive Science, a Perspective from Psychology and Artificial Intelligence. Allan Collins & Edward E. Smith, Morgan Kaufmann Publishers. San Mateo, California. pp. 312-322.
23. Schütze, H. 1997. Ambiguity Resolution in Language Learning: Computational and Cognitive Models. CSLI Lecture Notes 71, CSLI Publications, based on revised PhD Thesis, Stanford University, Dept. of Linguistics, July 1995.
24. Stevenson, M. (2002). Combining Disambiguation Techniques to Enrich an Ontology, Proceedings of the Fifteenth European Conference on Artificial Intelligence (ECAI-02)

workshop on "Machine Learning and Natural Language Processing for Ontology Engineering", Lyon, France.
25. Takayma, Y., Flournoy, R.S., Kaufmann, S. (1998). Information Mapping. Centre for the Study of Language and Information. Stanford University. Available at: www-csli.stanford.edu/semlab/infomap/Papers/takayama-infomap.ps
26. Tversky, B. (1989). Parts, Partonomies, and Taxonomies. Developmental Psychology, Vol. 25. No.6, pp 983–995.
27. Tversky, B, Hemenway, K. (1984). Objects, parts, and categories. Journal of Experimental Psychology: General, 113. pp. 169–193.
28. TWA sense tagged context data in http://www.cs.unt.edu/~rada/downloads.html
29. Widdows, D. (2004). Geometry and Meaning. CSLI Lecture Notes 172, CSLI Publications, Stanford University.
30. Wittengstein, L. (1953). Philosophical Investigations. MacMillan, New York.

A new proposal of Word Sense Disambiguation for nouns on a Question Answering System*

S. Ferrández[1], S. Roger[1,2], A. Ferrández[1], A. Aguilar[1], and P. López-Moreno[1]

[1] Natural Language Processing and Information Systems Group
Department of Software and Computing Systems
University of Alicante, Spain
[2] Natural Language Processing Group
Department of Computing Sciences
University of Comahue, Argentina
{sferrandez,sroger,antonio}@dlsi.ua.es,
{P.Lopez}@ua.es

Abstract. This paper describes the impact of the application of a Word Sense Disambiguation (WSD) algorithm for nouns on AliQAn [16], the Question Answering system with which we have participated in the CLEF-2005. Applying the traditional WSD decreases the performance in 4.7% on the Mean Reciprocal Rank (MRR). To solve this problem, we propose two different uses of WSD: (1) to choose a set of synsets instead of the traditional use of WSD, in which only one synset is chosen; (2) to disambiguate the words not present in EuroWordNet (EWN). Using our proposal of WSD the MRR increases a 6.3% with regard to the baseline without WSD. Furthermore, our proposal of WSD increases the MRR with regard to the traditional use of WSD in an 11%. Finally, the implementation of our approach of WSD is computationally efficient by means of a preprocessing of EWN.

1 Introduction

In this paper we analyze the benefits of a Word Sense Disambiguation (WSD) algorithm for nouns in AliQAn [16], a Spanish Question Answering (QA) system, with which we have participated in the CLEF-2005[1] competition.

QA objective consists of identifying the answer of questions in large collections of documents. QA is not a simple task of Information Retrieval (IR), QA tries to go beyond and returns a concrete answer in response to an arbitrary query. For the users, it is very interesting to find accurate information, thanks to the increment of available information. The QA systems are capable to answer questions formulated by the users in natural language.

* This research has been partially funded by the Spanish Government under project CICyT number TIC2003-07158-C04-01 and by the Valencia Government under project number GV04B-268.

[1] http://www.clef-campaign.org/

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 83-92

Current approaches to QA are mainly based on NLP tools or machine learning. There are different and possible implementations for QA systems. Generally, most of them are based on NLP tools [1, 2, 9, 13, 15], like Part of Speech (PoS) taggers, syntactic parsers, WSD, knowledge bases consisting of dictionaries, lexical-semantic data bases, ontologies and many others. Nevertheless, other systems use machine learning techniques with statistical models [5], such as Hidden Markov Models or Maximum Entropy. This is, in outline, the present situation.

The AliQAn system uses the NLP techniques. Our system has been developed during the last two years in the Department of Language Processing and Information Systems at the University of Alicante. It is based on complex pattern matching using NLP tools. Beside, WSD is applied to improve the system.

WSD algorithm is used in the phases of indexation and the search. In the first case, this algorithm allows disambiguation of the corpora words, and in the second one, it resolves ambiguities in the question words.

WSD has several critical problems. The running time of WSD algorithms makes difficult its use on huge corpora, as QA systems require. On the other hand, the low precision of WSD algorithms makes that this technique is not appropriated to be applied in QA systems. These two reasons do not allow to obtain interesting results applying WSD in real time QA systems. In order to solve these problems, we propose a concrete WSD algorithm that reduces its running time in 98.9%, due to a preprocess of EWN and improves the QA precision by means of: (1) selecting a set of synsets per word (instead of only one); (2) disambiguating words that are not presented in EWN.

The rest of the paper is organized as follows; section two describes the backgrounds of QA with regard to WSD; section three details AliQAn system with a brief description; section four explains our proposal of WSD algorithm on the AliQAn system; section five shows the evaluation results and finally, section six exposes our conclusions and discusses future works.

2 Backgrounds of QA with regard to WSD

Most of current monolingual QA systems [7, 10, 14, 18] do not apply any WSD algorithm. Nowadays, the use of WSD algorithms on QA and IR usually produce a decrease on the overall accuracy and an increase in time running.

Only in IR systems, the WSD techniques have been applied [8, 19, 17]. In the first analyzed system [8], the indexation with wordnet synsets improves the results of the IR system to 29% (from 30% up to 60%) but has the disadvantage that is carried out manually.

The project MEANING [19], has developed tools for the automatic acquisition of lexical knowledge that will help WSD. The obtained lexical knowledge is stored in the Multilingual Central Repository [4], which is based on the design of the EWN database. This implementation is based on the use of WSD with domains [12]. The problem using this technique is that the domains have to be

created in the previous phase. The observed precision in this system is 65.9%, which means an increase of 2% (Table 1, System A) with WSD.

Finally, the last system [17], uses a combination of high precision techniques and sense frequency statistics in an attempt to reduce the impact of erroneous disambiguation on retrieval performance. The incremented precision in the final result is 1.73% over 62.1% (Table 1, System B).

Table 1. Different IR systems results with the obtained improvements once WSD is applied

	System A	System B
Without WSD	65.9%	62.1%
WSD Improvement	2%	1.73%

3 The AliQAn system

AliQAn (Figure 1), is a monolingual open-domain QA system based on the intensive use of NLP tools. AliQAn has participated in the Spanish QA CLEF-2005 competition [16], in which it was ranked third.

The AliQAn architecture is divided in two main phases: Indexation phase and Search phase. In both phases, both in the document corpora and the questions, the same NLP process is applied: PoS tagging, partial parsing and WSD.

In order to make the syntactic analysis, SUPAR [6] system is used, which works in the output of MACO [1] PoS tagger. Beside, WSD is applied using EWN.

AliQAn identifies the different grammatical structures of the sentence, named syntactic blocks (SB). This is realized using the output of SUPAR, which performs partial syntactic analysis. These blocks are verb phrases (VP), nominal phrases (NP) or prepositional phrases (PP).

For example in the sentence: *"Kim Il Sung died at the age of 80"*, the obtained list of SB is: [NP,kim∗il∗sung][VP,to die][PP, at: age [PP, of: 80]].

Indexation phase, the first phase of AliQAn, consists of arranging the data where the system tries to find the answer of the questions. Two different indexation are carried out: IR-n [11] and QA indexation.

In the second phase, the search, the following three tasks are sequentially performed: question analysis, selection of relevant passages and extraction of the answer.

In the first task the system detects the type and the case of the question. The WordNet Based-Types and EWN Top-Concepts have been considered for the question type, the question case determines the set of syntactic patterns to use in the extraction of the answer.

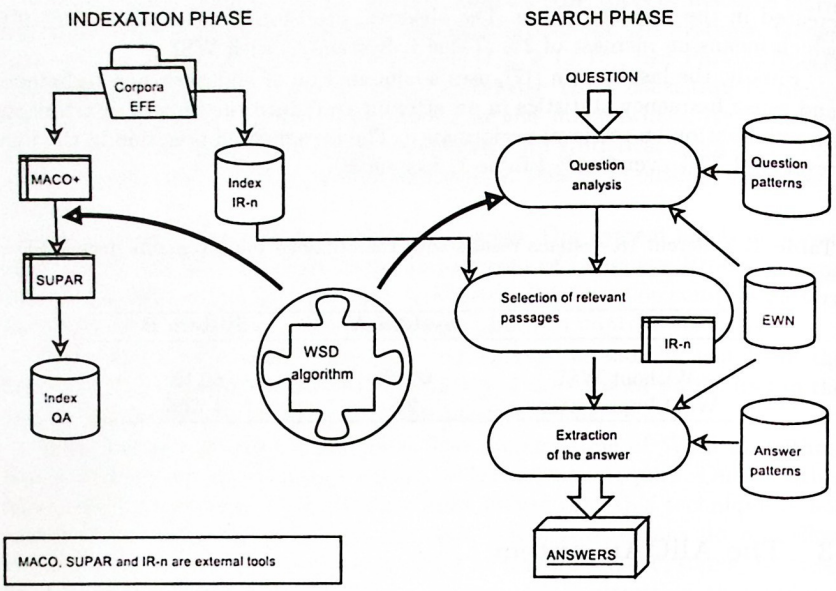


Fig. 1. AliQAn system architecture

Besides, the selection of the question terms (keywords) is carried out. These keywords are used in the next task, the selection of relevant passages, which is developed by the IR-n [11] system. IR-n returns a list of passages where the system applies the extraction of the answer, the last task of AliQAn, where it tries to extract the correct answer to the question using the syntactic patterns.

Next, an example (question 114, *In Workshop of Cross-Language Evaluation Forum (CLEF 2003)*) of resolution of one question, where the system chooses the correct solution.

- **Question:** A qué primer ministro abrió la Fiscalía de Milán un sumario por corrupción? (To whom prime minister the Office of the public prosecutor of Milan opened a summary for corruption?)
- **Type:** person
- **Case:** 3
- **List of SB:**
  - **NP1:** ([NP, primer*ministro])
  - **VP:** ([VP, abrir])
  - **NP2:**([NP, fiscalia [PP, de: milan]])([NP, sumario [PP, por: corrupcion]])
- **Text where the correct solution is:** "[...] la **Fiscalía de Milán abrió**, hoy martes, un **sumario** al **primer ministro**, *Silvio Berslusconi*, por un supuesto delito de **corrupción** [...]"
- **Text where the incorrect solution is:** "[...] **primer ministro** y líder socialista, *Bettino Craxi*, al que el pasado 21 de septiembre Paraggio **abrió** un **sumario** relacionado con el proyecto Limen por supuestos delitos de **corrupción** [...]"
- **Answer:** Silvio Berlusconi

4 The WSD algorithm

The WSD algorithm analyzed in this paper is the one proposed by Agirre and Rigau [3]. This WSD algorithm is based on the use of conceptual distance in order to try to provide a basis for determining the closeness in meaning among words, taking as reference the hierarchical structure of EWN. The conceptual distance is captured by a Conceptual Density (CD) formula. Given a concept c, the CD is calculated using the next equations:

$$CD(c,m) = \frac{\sum_{i=0}^{m-1} nhyp^i}{descendants_c} \quad (1)$$

$$descendants_c = \sum_{i=0}^{h-1} nhyp^i \quad (2)$$

where $nhyp$ is number of hyponyms, m is the number of marks of words senses and h is the height of the subhierarchy.

Unlike the algorithm proposed by Agirre and Rigau [3], our implementation selects not just one synset, but a set of the most probable synsets. This is because the algorithm attempts to discard only completely wrong synsets (keeping related synsets) in order to improve the precision of the system. Moreover, the proper nouns that are not in EWN are disambiguated too with respect to the following synsets: 05369359 (person) 07451540 (object) 08229827 01218276 (place).

The behavior of our implementation is shown in the next example:

- In the sentence
  - *El presidente de Guinea, Obiang, sugirió hoy, viernes, que su Gobierno podría rechazar la ayuda internacional* (The president of Guinea, Obiang, suggested today, Friday, that his Government could refuse the international help)
- Synsets for the word *"presidente"* sorted by [3]
  - 09400170 006140480 11176111 01090427 08956043
- Correct synset of the word *"presidente"* in this sentence
  - 00614048
- Synset returned by Agirre tool [3]
  - 09400170
- Set of synsets selected of the word *"presidente"* using our implementation
  - 09400170 00614048

In this example, the word *"Obiang"* that is not in EWN is also disambiguated. In this case, Agirre tool [3] returns the right synset: 05369359 (person).

Our proposals allow to improve the precision of our QA system as the evaluation section will show, because the WSD algorithms usually have a low precision (about 66%). It makes quite probable to discard the right synset, which is worst than keeping all the synsets of the word. In this way, our proposal only discards completely wrong synsets, which are not used when the question terms are compared with the document terms. For example, in the question "¿Cuántos muertos al año causan las minas antipersona en el mundo? (How many deaths

per year are caused by the anti-personnel mines in the world?)", the document sentence that speaks about a "coal mine" will not be processed. Moreover, our proposal overcomes the disambiguation that uses EWN domains [19] because it does not require any grouping of EWN, which could also introduce errors in the system.

The next example shows the improvement obtained when our WSD implementation is used.

- For the question:
  - *Quién es el director de la CIA?* (Who is the director of CIA?)
- Without using disambiguation, AliQAn returns:
  - *Servicio Central de Información* (Information Central Service)
  - *Servicio Central de Inteligencia* (Intelligent Central Service)
  - *Servicio Central de Información de EEUU* (Information Central Service of USA)
- Using disambiguation, AliQAn returns:
  - William Colby
  - Robert Gates
  - James Woolsey

The Agirre and Rigau [3] WSD algorithm presents a precision of 66%. Our WSD implementation of this algorithm has been evaluated in the EFE corpora (detailed in the evaluation section), where its precision stays at 60% when we select only one synset. When a 40% of the synsets of a word is selected, its precision stays at 76%. Regarding to the precision of our proposal for nouns that are not in EWN, the precision stays at 65%. Finally, the running time of the algorithm has been reduced by means of storing all the required information shown in equations (1) and (2), such as the number of descendants, for each synset in EWN. In this way, the running time has been reduced in a 98.9% (from 1400 seconds to 15 seconds in disambiguating 9 files).

5 Evaluation

5.1 Dataset

The experiments described in this section have been carried out using the AliQAn system with CLEF 2003 questions and corpora, i.e. Spanish corpora EFE 1994. The collection, which was indexed by the IR-n system, contains approximately 215.738 documents for a total of 509 Megabytes.

The our test set has been extracted from the CLEF 2003 competition task, and it includes a total of 200 questions. The type of questions capable to contain noun as answer are: definition, abbreviation, event and person. This corresponds to the 65% of all questions and the 35% remaining corresponds to the following type of questions: date, month, percentage, object, quantity, economic, age, measure, period, year. Finally, the average ambiguity factor for each term in EuroWordNet 1.6 is ZZZ

5.2 Evaluation Measures

In order to be able to evaluate the system we need a measure that values the general results of the system. Mean Reciprocal Rank (MRR) is the measure used in the CLEF 2003 campaign for evaluating the systems.

$$MRR = (\sum_{i=1}^{Q} \frac{1}{far(i)})/Q$$

where Q is the number of questions (200 in our case) and $far(i)$ indicates the position of the first correct answer. The value of $1/far(i)$ will be 0 if the system has not found the answer.

5.3 Results analysis

The architecture and behavior of our system have been described in previous sections. Now we are going to present the obtained results and the study about the performance of the system when WSD is used in the corpora and questions.

The comparison will be performed using the AliQAn system with three algorithms, which have different WSD levels. These algorithms are shown in Table 2.

1. **Baseline without WSD:** (first column of the Table 2) The baseline does not have WSD algorithm, i.e. all the synsets per word are used. This base system has got a MRR of 44.5%.
2. **1-Sense WSD algorithm:** (second column of the Table 2). In this case, the WSD algorithm used is the propose by Agirre et al. [3]. This algorithm chooses one synset for disambiguating a word and the MRR obtained is 42.4%.
3. **Our proposal of WSD algorithm:** (third column of the Table 2). In this case, the set of most probable synsets is chosen. The selected synsets are the 40% of total word synsets. The system has got a MRR of 47.3%.

Table 2. Results using different applications of WSD on 200 questions of CLEF 2003

	1 Baseline without WSD	2 1-Sense WSD	3 Our proposal of WSD
MRR	44.5	42.4 (-4.75%)	47.3 (+6.3%)
% of First Correct Answer (FCA)	39	37.5 (-3.85%)	42.5 (+8.97%)
FCA Improvement (200 questions)		3 questions	7 questions
		+11% of improvement in the MRR with regard to the 1-Sense WSD	

Therefore, our WSD proposed has increased the MRR compared to the 1-sense WSD algorithm up to 11% and a 6.3% regarding the baseline (see Figure 2), it increases the number of queries answered in first place in 7.

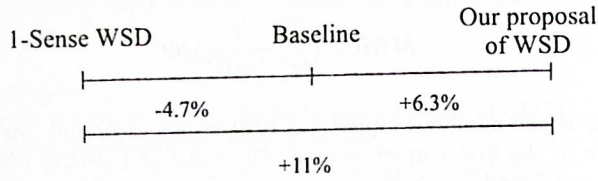


Fig. 2. Relationship with regard to percentages of the WSD system presented in Table 2

In comparison with the 42.4 of the 1-sense WSD algorithm, the percentage of increment in the MRR is 8.3% when a 10% of the synsets is selected. It is 9.2% when a 20% is selected. It is 10.4% when a 30% is selected. Finally, it is 11% when a 40% is selected.

We have carried out an analysis of the number of the disambiguated nouns in the 200 questions (159,778), where 46,194 nouns have only one synset in WordNet, and the remaining nouns (113,584) have 3.9 synsets in WordNet on average. Our proposal of WSD selects 1.9 synsets on average (in case of draw in the score between 2 synsets, both are selected). Moreover, 44,247 nouns that are not in WordNet have been disambiguated.

On the other hand, normally the time of carrying out the corpus processing WSD is too long or at worst NP complete. Many researchers have been confronted with this problem when they apply their approaches of WSD, for this reason they only apply WSD to questions. With our WSD algorithm, we have achieved to reduce considerably the running time required. Initially, we needed 1400 seconds for the processing of 9 files, now we do it within 15 seconds, which supposes a decrease of 98.9%.

6 Conclusions and future works

In this paper we propose an algorithm that aims at enhancing WSD for nouns on a QA system. This algorithm is based on the algorithm proposed by Agirre et al. [3]. The difference is that while it considers one synset for disambiguating a word, our proposal selects the most relevant synsets and adds the disambiguation of the proper nouns that are not included in EWN 1.6. In order to evaluate our algorithm, a number of comparisons have been carried out. Results confirm the viability of our algorithm, showing an improvement up to 11% over the traditional WSD algorithm and the 6.3% over the baseline. Furthermore, we have greatly reduced the computational cost for WSD process by 98.9% by means of a

preprocessing of EWN. It is important to emphasize that the algorithm proposed does not require any previous grouping (such as domains) for the disambiguation process, which may involve some error.

The contribution of our paper to the WSD research area is that the traditional 1-sense WSD algorithms do not improve QA, as it is stated in Table 2 and Figure 2 (-4.7% in the MRR). That is because of their low precision. Our proposal allows selecting a percentage of synsets instead of only one, as the traditional 1-sense WSD algorithms do. In this way, the MRR of a 1-sense WSD algorithm is improved in 11%, and it improves the MRR of a QA system without WSD in 6.3%. Moreover, our proposal overcomes the traditional drawback of WSD that is its high computational cost, which makes too difficult its application to huge corpora. Our approach reduces the running time of the WSD algorithm in a 98.9%.

The results are promising. Therefore we expect to analyze the results on CLEF 2004 and CLEF 2005. In the future, we are going to develop other WSD algorithm to prove that our proposal is validated independently of the WSD algorithm itself.

References

1. S. Acebo, A. Ageno, S. Climent, J. Farreres, L. Padró, R. Placer, H. Rodriguez, M. Taulé, and J. Turno. MACO: Morphological Analyzer Corpus-Oriented. *ESPRIT BRA-7315 Aquilex II, Working Paper 31*, 1994.
2. A. Ageno, D. Ferrés, E. González, S. Kanaan, H. Rodríguez, M. Surdeanu, and J. Turmo. TALP-QA System for Spanish at CLEF-2004. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, pages 425 – 434, 2004.
3. E. Agirre and G. Rigau. A proposal for word sense disambiguation using conceptual distance. *1st Intl. Conf. on recent Advances in NLP. Bulgaria*, 1995.
4. J. Atserias, L. Villarejo, G. Rigau, E. Agirre, J. Carrol, and B. Magnini P. Vossen. The meaning multilingual central repository. *In Proceeding of the Second International WordNet Conference-GWC.*, pages 23–30, 2004.
5. C. de Pablo, J.L. Martínez-Fernández, P. Martínez, J. Villena, A.M. García-Serrano, J.M. Goñi, and J.C. González. miraQA: Initial experiments in Question Answering. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, pages 371 – 376, 2004.
6. A. Ferrández, M. Palomar, and L. Moreno. An Empirical Approach to Spanish Anaphora Resolution. *Machine Translation. Special Issue on Anaphora Resolution In Machine Translation*, 14(3/4):191–216, December 1999.
7. J. M. Gómez, E. Bisbal, D. Buscaldi, and P. Rosso E. Sanchis. Monolingul and Cross-Language QA using a QA-oriented Passage Retrieval System. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
8. J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarrán. Indexing with wordnet synsets can improve text retrieval. *In the Proceedings of the ACL/COLING Workshop on Usage of WordNet for Natural Language Processing*, pages 38–44, 1998.
9. J. Herrera, A. Peñas, and F. Verdejo. Question Answering Pilot Task at CLEF 2004. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, pages 445 – 452, 2004.

10. D. Laurent, P. Séguéla, and S. Negre. Cross Lingual Question Answering using QRISTAL for CLEF 2005. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
11. F. Llopis and J.L. Vicedo. Ir-n, a passage retrieval system. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, 2001.
12. Bernardo Magnini, Carlo Strapparava, Giovanni Pezzulo, and Alfio Gliozzo. The role of domain information in word sense disambiguation. *Nat. Lang. Eng.*, 8(4):359–373, 2002.
13. E. Méndez-Díaz, J. Vilares-Ferro, and D. Cabrero-Souto. COLE at CLEF 2004: Rapid prototyping of a QA system for Spanish. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, pages 413 – 418, 2004.
14. G. Neumann and B. Sacaleanu. DFKI's LT-lab at the CLEF 2005 Multiple Language Question Answering Track. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
15. M. Pérez-Coutiño, T. Solorio, M. Montes y Gómez, A. López-López, and L. Villaseñor-Pineda. The Use of Lexical Context in Question Answering for Spanish. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, pages 377 – 384, 2004.
16. S. Roger, S. Ferrández, A. Ferrández, J. Peral, F. Llopis, A. Aguilar, and D. Tomás. AliQAn, Spanish QA System at CLEF-2005. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, 2005.
17. Christopher Stokoe, Michael P. Oakes, and John Tait. Word sense disambiguation in information retrieval revisited. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 159–166, New York, NY, USA, 2003. ACM Press.
18. R. F. E. Sutcliffe, M. Mulcahy, and I. Gabbay. Cross-Language French-English Question Answering Using the DLT system at CLEF 2005. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
19. P. Vossen, G. Rigau, I. Alegria, E. Agirre, D. Farwell, and M. Fuentes. Meaningful results for information retrieval in the meaning project. *Third International WordNet Conference*, 2006.

On Types, Intension and Compositionality

Walid S. Saba

Department of Computer Science
American University of Technology (AUT)
Byblos-Highway, Byblos, Lebanon P. O. Box 20
walid.saba@aut.edu

Abstract. In this paper we demonstrate that a number of challenging problems in the semantics of natural language, namely the treatment of the so-called intensional verbs and the semantics of nominal compounds, can be adequately resolved in the framework of compositional semantics, if a strongly-typed ontological structure is assumed. In addition to suggesting a proper treatment of nominal compounds and intensional verbs within the framework of compositional semantics, we briefly discuss the nature of this ontological type system and how it may be constructed.

1 The Semantics of Nominal Compounds

The semantics of nominal compounds have received considerable attention by a number of authors, most notably (Kamp & Partee, 1995; Fodor & Lepore, 1996; Pustejovsky, 2001), and to our knowledge, the question of what is an appropriate semantics for nominal compounds has not yet been settled. In fact, it seems that the problem of nominal compounds has presented a major challenge to the general program of compositional semantics in the Montague (1973) tradition, where the meaning of a compound nominal such as $[N_1\ N_2]$ is generally given as follows:

(1) $\|N_1\ N_2\| = F(\|N_1\|, \|N_2\|)$

In the simplest of cases, the compositional function F is usually taken to be a conjunction (or intersection) of predicates (or sets). For example, assuming that red(x) and apple(x) represent the meanings of red and apple, respectively, then the meaning of a nominal such as red apple is usually given as

(2) $\|\text{red apple}\| = \{x \mid \text{red}(x) \wedge \text{apple}(x)\}$

What (2) says is that something is a red apple if it is red and apple. This simplistic model, while seems adequate in this case (and indeed in many other instances of

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 93-102

similar ontological nature), clearly fails in the following cases, all of which involve an adjective and a noun:

(3) former senator
(4) fake gun
(5) alleged thief

Clearly, the simple conjunctive model, while seems to be adequate for situations similar to those in (2), fails here, as it cannot be accepted that something is former senator if it is former and senator, and similarly for (4) and (5). Thus, while conjunction is one possible function that can be used to attain a compositional meaning, there are in general more complex functions that might be needed for other types of ontological categories. In particular, what we seem to have is something like the following:

(6) $\|\text{red apple}\| = \{x | x \text{ is red and } x \text{ is apple}\}$
(7) $\|\text{former senator}\| = \{x | x \text{ was but is not now a senator}\}$
(8) $\|\text{fake gun}\| = \{x | x \text{ looks like but is not actually a gun}\}$
(9) $\|\text{alleged thief}\| = \{x | x \text{ could possibly turn out to be a thief}\}$

It would seem, then, that different ontological categories require different compositional functions to compute the meaning of the whole from the meanings of the parts. In fact, the meaning (intension) of some compound might not be captured without resorting to temporal and/or modal operators. This has generally been taken as an argument against compositionality, in that there does not seem to be an answer as to what the compositional semantic function F in $\|N_1\ N_2\| = F(\|N_1\|, \|N_2\|)$ might be. We believe, however, that this is a fallacious argument in that the problem is not due to compositionality but in 'discovering' a number of semantic functions that could account for all nominal compounds of different ontological categories. Moreover, we believe that the answer lies in assuming a richer type structure than the flat type system typically assumed in Montague-style semantics.

2 Ontology and the Semantics of Adjectives

In (2) we stated that the meaning of some adjectives. The question however is what "kinds" of adjectives are specifically intersective. It would seem that for constructions of the form [A N] where A is a physical property (such as red, large, heavy, etc.) and N is a object of type PhysicalThing (such as car, person, desk, etc.), the meaning of [A N] can be obtained as follows:

(10) $\|A\ N\| = \{x | A_{PhysicalProperty}(x) \wedge N_{PhysicalThing}(x)\}$

Note here that the above expression is not a statement about the meaning of any particular adjective. Instead, what (10) simply states is that some adjectives, such as large, heavy, etc. are intersective. Thus, in $\|\text{large table}\| = \{x | \text{large}(x) \wedge \text{table}(x)\}$.

for example, it is assumed that the meaning of large, namely the predicate large(x) has been defined. Although the semantics of such adjectives is not our immediate concern here, it must be pointed out that semantics of such (intersective) adjectives, which are presumably the simplest, can be quite involved, as these adjectives are very context-sensitive – clearly the sense of 'large' in 'large elephant' is quite different from the sense of 'large' in 'large bird'. Assuming a predicate $typical_{a,T}(x)$, which is true of some object x of type T if x is a typical object with respect to one of its attributes a is defined, then the meanings of such adjectives as large and heavy, for example, could be defined as follows, where $x :: T$ refers to an object x of type T:

(11) $large \Rightarrow (\forall x :: PhysicalThing)(large(x) \equiv_{df}$
$\lambda P[P(x) \wedge (\exists y :: PhysicalThing)(P(y) \wedge typical_{size}(y)$
$\wedge size(x,s_1) \wedge size(y,s_2) \wedge (s_1 >> s_2))])$

(12) $heavy \Rightarrow (\forall x :: PhysicalThing)(heavy(x) \equiv_{df}$
$\lambda P[P(x) \wedge (\exists y :: PhysicalThing)(P(y) \wedge typical_{weight}(y)$
$\wedge weight(x,w_1) \wedge weight(y,w_2) \wedge (w_1 >> w_2))])$

What (1) and (2) say is the following: that some P object x, is a large (heavy) P, iff it has a size (weight) which is larger than the size (weight) of another P object, y, which has a typical size (weight) as far as P objects go. It would seem, then, that the meaning of such adjectives is tightly related to some attribute (large/size, heavy/weight, etc.) of the corresponding concept. Thus, such adjectives, while they are intersective, are context-dependent: their meaning is fully specified only in the context of a specific concept.

One of the main points that we like to make in this paper is that, like intersective adjectives, non-intersective adjectives also have a compositional meaning, although the compositional function might be more involved than simple conjunction. For example, we argue that the following are reasonable definitions for fake, former and alleged:

(13) $(\forall x :: \mathbf{PhysicalArtifact})(fake(x) \equiv_{df}$
$\lambda P[(\exists y :: \mathbf{Physical})(\neg P(x) \wedge P(y) \wedge similar_{\{shape,size\}}(x,y))]$

(14) $(\forall x :: \mathbf{Role})(former(x) \equiv_{df}$
$\lambda P[(\exists t)((t < now) \wedge P(x,t) \wedge \neg P(x,now))])$

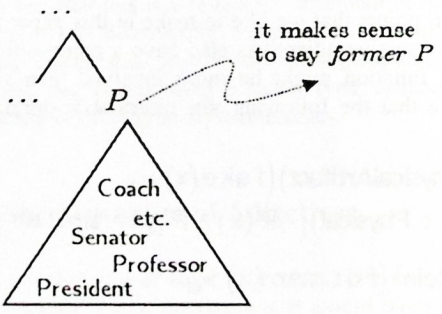
(15) $(\forall x :: \mathbf{Role})(alleged(x) \equiv_{df}$
$\lambda P[(\exists t)((t > now) \wedge \neg P(x,now) \wedge \Diamond P(x,t))]$

That is, 'fake' applies to some concept P as follows: a certain physical object x is a fake P iff it is not a P, but looks like (in certain respects) to something else, say y, which is actually a P. On the other hand, what (14) says is the following: a

certain x is a former P iff x was a P at some point in time in the past and is not now a P. Finally, what (15) says is that something is an 'alleged' P iff it is not now known to be a P, but could possibly turn out to be a P at some point in the future.

It is interesting to note here that the intension of fake and that of former and alleged was in one case represented by recourse to possible worlds semantics (the case of (14) and (15)), while in (13) the intension uses something like structured semantics, assuming that $similar_{\{A_1, A_2, \ldots A_n\}}(x, y)$ which is true of some x and some y if x and y share a number of important features, is defined. What is interesting in this is that it suggests that possible-worlds semantics and structured semantics are not two distinct alternatives to representing intensionality, as has been suggested in the literature, but that in fact they should co-exist.

Additionally, several points should also be made here. First, the representation of the meaning of fake given in (13) suggests that fake expects a concept which is of type PhysicalArtifact, and thus something like fake idea, or fake song, for example, should sound meaningless, from the standpoint of commonsense[1]. Second, the representation of the meaning of former given in (14) suggests that former expects a concept which has a time dimension, i.e. is a temporal concept. Finally, we should note here that our ultimate goal of this type of analysis is to discover the ontological categories that share the same behavior. For example, an analysis of the meaning of former, given in (14), suggests that there are a number of ontological categories that seem to share the same behavior, and could thus replace P in (14), as implied by the fragment hierarchy below.



3 Types, Predicates and Logical Concepts

In "Logic and Ontology" Cocchirarella (2001) argues for a view of logic as a language in contrast with the view of logic as a language. In the latter, logic is

[1] One can of course say fake smile, but this is clearly another sense of fake. While fake gun refers to a gun (which is of type Artifact) that is not real, fake smile refers to a dishonest smile, or a smile that is not genuine.

viewed as an "abstract calculus that has no content of its own, and which depends on set theory as a background framework by which such a calculus might be syntactically described and semantically interpreted." In the view of logic as a language, however, logic has content, and "ontological content in particular." This view however necessitates the use of type theory, as opposed to set theory as the background framework. It is this view that we advocate here, and in our opinion, problems in the semantics of natural language cannot be resolved until a logic that is grounded in type theory and predication (as opposed to set membership) is properly formulated. In this section we discuss the building blocks of such a program.

3.1 Types vs. Predicates

In formal (programming) languages we write statements such as 'int x', which is a type declaration statement meaning that x is an object of type int. However, in programming languages we do not have procedures that verify (somehow) if some object is of a certain type - that is we do not have a predicate such as int (x) that takes some object x and returns 'true' if x is an int and 'false' otherwise. Clearly, the type and the corresponding predicate are related, and in particular, a predicate such as int (x) is true of some object x if has all the properties of the type int.

Like objects in formal (programming) languages, commonsense objects have a type, and a corresponding predicate that verifies if a certain object is of a specific type. For instance, our ontology has a type hierarchy that contains the following fragment:

(16) $\mathbf{Piano} \supset \mathbf{Instrument} \supset \ldots \supset \mathbf{Artifact} \supset \ldots \supset \mathbf{PhysicalThing} \supset \mathbf{Thing}$

Corresponding to these types there are predicates such as piano(x), instrument(x), etc. Moreover, a predicate such as piano(x) is true of some object x just in case x happens to be a piano. That is, such concepts correspond to what Cocchirarella (2001) refers to as 'first intentions', i.e., concepts abstracted directly from physical reality. The point here is that what makes some object x a piano, for example (or, what makes piano(x) true of some object x) is determined directly from physical reality. Such 'first intentions' concepts should be contrasted with concepts that are about 'second intentions', which, according to Cocchirarella are "concepts abstracted wholly from the 'material' content of first intentions", using the logical apparatus. Thus first intention concepts are in some sense 'ontological concepts', while second intention concepts can be thought of as 'logical concepts'.

Continuing with our example, piano(x) would be an ontological concept, while pianist(x), for example, is a concept that is logically defined using the concept piano(x), and perhaps other 'first intention' concepts. In other words, what makes pianist(x) true of some x is not physical reality but some set of logical conditions. This can be stated as follows:

(17) $(\forall x :: \mathbf{Artifact})(\text{piano}(x) \equiv_{df} \text{NueralNetPatternRecogProc}(x))$

(18) $(\forall x :: \textbf{Human})(\text{pianist}(x)$
$\equiv_{df} (\exists a :: \textbf{Activity})(\exists p :: \textbf{Artifact})(\text{playing}(a) \wedge \text{piano}(p)$
$\wedge \text{agent}(a,x)$
$\wedge \text{object}(a,\textbf{Music})$
$\wedge \text{instrument}(a,p))$

That is, something is a piano if it looks like, sounds like, feels like, etc. what we call 'piano'. On the other hand, what (18) says is the following: any Human x, is a pianist, iff there is some playing activity and some Artifact which is a piano where the object of this playing activity is Music and the instrument of this activity is a piano.

3.2 Compound Nominals Revisited

The problem of compound nominals in the case of noun-noun combinations has traditionally been due to the various relations that are usually implicit between the nouns (see Weiskopf, forthecoming). For example, consider the following:

(19) $\|\text{brick house}\| = \{x | x \text{ is a house that is } \textbf{made of} \text{ brick}\}$
(20) $\|\text{dog house}\| = \{x | x \text{ is a house that is } \textbf{made for} \text{ a dog}\}$
(21) $\|\text{beer drinker}\| = \{x | x \text{ often drinks beer}\}$
(22) $\|\text{beer factory}\| = \{x | x \text{ is a factory that } \textbf{makes} \text{ beer}\}$

Thus, while a brick house is a house 'made of' bricks, a dog house is a house that is 'made for' a dog. It would seem, then, that the relation implicitly implied between the two nouns differ with different noun-noun combinations. However, assuming the existence of a strongly-typed ontology might result in identifying a handful of implicit relations that can account for all patterns. Consider for example the following:

(23)
$\|\text{brick house}\| = \{x : \textbf{Artifact} | \text{house}(x) \wedge (\exists y : \textbf{Substance})(\text{brick}(y) \wedge \text{madeOf}(x,y))\}$

(24)
$\|\text{paper cup}\| = \{x : \textbf{Artifact} | \text{cup}(x) \wedge (\exists y : \textbf{Substance})(\text{paper}(y) \wedge \text{madeOf}(x,y))\}$

(25)
$\|\text{plastic knife}\| = \{x : \textbf{Artifact} | \text{knife}(x) \wedge (\exists y : \textbf{Substance})(\text{plastic}(y) \wedge \text{madeOf}(x,y))\}$

It would seem, therefore, that the same semantic relation, namely madeOf, is the relation that is implicit between all $[N_1\ N_2]$ combinations when N_1 is an Artifact and N_2 is a Substance. Similarly, it would seem that the same semantic relation underlies all $[N_1\ N_2]$ combinations when N_1 is a Human and N_2 is a Substance, where P should be read as 'it is often the case that P', or 'generally, P':

(26) $\|\text{beer drinker}\| = \{x :: \textbf{Human} | (\exists y :: \textbf{Substance})(\text{beer}(y) \wedge \Delta(\text{drinks}(x,y)))\}$

(27) $\|\text{cigar smoker}\| = \{x :: \textbf{Human} | (\exists y :: \textbf{Substance})(\text{cigar}(y) \wedge \Delta(\text{smokes}(x,y)))\}$

4 The So-Called Intensional Verbs

In (Montague, 1969) Montague discusses a puzzle pointed out to him by Quine which can be illustrated by the following examples:

(16) $\|\text{John painted a unicorn}\| = (\exists x)(\text{unicorn}(x) \wedge \text{painted}(j,x))$

(17) $\|\text{John found a unicorn}\| = (\exists x)(\text{unicorn}(x) \wedge \text{found}(j,x))$

The puzzle Quine was referring to was the following: both translations admit the inference $(\exists x)(\text{unicorn}(x))$ – that is, both sentences imply the existence of a unicorn, although it is quite clear that such an inference should not be admitted in the case of (17). According to Montague, the obvious difference between (16) and (17) must be reflected in an ontological difference between find and paint in that the extensional type $(e \rightarrow (e \rightarrow t))$ both transitive verbs are typically assigned is too simplistic. Montague was implicitly suggesting that a much more sophisticated ontology (i.e., a more complex type system) is needed, one that would in fact yield different types for find and paint. One reasonable suggestion for the types of find and paint, for example, could be as follows:

(18) $\text{find} :: (e_{\textbf{Animal}} \rightarrow (e_{\textbf{Thing}} \rightarrow t))$

(19) $\text{paint} :: (e_{\textbf{Human}} \rightarrow (e_{\textbf{Representation}} \rightarrow t))$

Thus instead of the flat type structure implied by $(e \rightarrow (e \rightarrow t))$, the types of find and paint should reflect our commonsense belief that we can always speak of some **Animal** that found something (i.e., any **Thing** whatsoever), and of a **Human** that painted some illustration, or as we called it here a **Representation**. Before we proceed, however, we point out that throughout, we will use this **Font** for concept types in the ontology, and this font for predicate names. Thus, **x::LivingThing** means x is an object/entity of type LivingThing and apple(x) means the predicate or property apple is true of x. Note, further, that in a flat-type system, the expression $(\exists x)(\text{unicorn}(x) \wedge \text{found}(j,x))$ is equivalent to the typed expression $(\exists x :: \textbf{Entity})(\text{unicorn}(x) \wedge \text{found}(j :: \textbf{Entity}, x))$ since in flat type system there is only one type of entity. With this background, the correct translations of (18) and (19) and the corresponding inferences can now be given as follows:

(20) (∃x:**Thing**) (unicorn (x) ∧ found (j:**Rational**,x))
⇒ (∃x:**Thing**) (unicorn (x))
⇒ (∃x:**Thing**) (found (j:**Rational**,x))

(21) (∃x:Representation) (unicorn (x) ∧ painted (j:Rational,x))
⇒ (∃x:Representation) (unicorn (x))
⇒ (∃x:Representation) (painted (j:Rational,x))

Adding a rich type structure to the semantics, it seems, provides a reasonable solution to Quine's puzzle, as the correct inferences can now be made: if John found a unicorn, then one can indeed infer that an actual unicorn exists[2]. However, the painting of a unicorn only implies the existence of a representation (an illustration) of something we call a unicorn! Stated yet in other words, (7) implies that a unicorn Thing (including perhaps a unicorn Toy) exists, while (8) implies a unicorn Representation exists. There are two points that this discussion intends to emphasize: (i) is the need for a rich type structure to solve a number of problems in the semantics of natural language; and (ii) that this type structure is actually systematically discovered by an analysis of how ordinary language is used to talk about the world.

5 Language, Logic, Ontology and Commonsense

Our work here has been motivated by the (rather strong) claim of Richard Montague (see the paper on ELF in (Thomasson, 1974)) that there is no theoretical difference between formal and natural languages. If does turn out that Montague is correct (as we believe to be the case), then there should exist a formal system, much like arithmetic, or any other algebra, for concepts, as has been advocated by a number of authors, such as Cresswell (1973) and Barwise (1989), among others. What we are arguing for here is a formal system that explains how concepts of various types combine, forming more complex concepts in a formal, strongly-typed system. To illustrate, consider the following:

(22) artificial :: NaturalKind → Artifact
(23) flower :: Plant
(24) flower :: Plant ⊃ LivingThing
(25) flower :: Plant ⊃ LivingThing ⊃ NaturalKind
(26) artificial flower :: Artifact

What the above says is the following: artificial is a function that takes a NaturalKind and returns an Artifact (22); a flower is a Plant (23); a flower is a Plant which in turn is a LivingThing (24); a flower is a Plant, which is a LivingThing, which in turn is a NaturalKind (25); and, finally, an artificial flower is an Artifact (26). Therefore, 'artificial c', for some NaturalKind c, should in the final analysis have the same properties that any other Artifact has. Thus, while a flower, which is of type Plant, and is therefore a LivingThing, grows, lives and dies like any other LivingThing, an

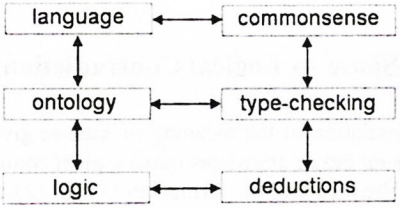
[2] Of course, in such a type system we would have Rational ⊃ Animal and therefore John, an entity of type Rational, can be the subject of found which expects an entity of type Animal.

artificial flower, and like any other **Artifact**, is something that is manufactured, does not grow, does not die, but can be assembled, destroyed, etc. The concept algebra we have in mind should also systematically explain the interplay between what is considered commonsense at the linguistic level, type checking at the ontological level, and deduction at the logical level. For example, the concept artificial car, which is a meaningless concept from the standpoint of commonsense, is ill-typed since **Car** is an **Artifact**, and **Artifact** does not unify with **NaturalKind** – neither type is a sub-type of the other.

The concept former father, on the other hand, which is also a meaningless concept from the standpoint of commonsense, escapes type-checking since father, which is a **Role**, is a type that former expects as shown in (29) below.

(29) former :: **Role** → **Role**

However, although former father escapes type-checking, the fact that this a meaningless concept from the standpoint of commonsense, is ultimately detected at the logical level by resulting in a contradiction as shown in the appendix. Thus what is meaningless at the linguistic level should be flagged at the type-checking level, or, if happens to escapes type-checking, such as former father, it should eventually result in a logical contradiction at the logical level (see the appendix concerning former father). The picture we have in mind can therefore be summarized as shown in the figure below.



6 Concluding Remarks

A number of problems in the semantics of natural language can be resolved in a compositional semantic framework if a rich type system that models an ontology of commonsense concepts can be assumed. If this where to happen, it would mean that there is a formal system that underlies natural language and that a concept algebra must exist. This subsequently means that the ontology we have in mind must be systematically discovered and cannot be invented, as has been argued by Saba (2001). In this paper we have shown that assuming such a rich type systems can help resolving a number of challenging problems in the semantics of natural language. For lack of space, in this paper we could not discuss the nature of this ontological structure, the corresponding strongly-typed meaning algebra, and how this structure might be discovered rather than invented, using natural language itself as a guide in this process. Some of these issues are discussed in some detail in Saba (2006).

References

1. Barwise, J. (1989), The Situation in Logic, CSLI Publications, Stanford.
2. Cocchiarella, N. B. (2001), Logic and Ontology, Axiomathes 12, pp. 117-150.
3. Cresswell, M. J. (1973), Logics and Languages, Methuen & Co., London.
4. Kamp, H. & B. Partee. 1995. Prototype theory and Compositionality, Cognition 57, pp. 129–191.
5. Montague, R., 1960. On the Nature of Certain Philosophical Entities, The Monist 53, pp. 159–194
6. Montague, R., 1974. In Thomason, R. (1974) (Ed.), Formal Philosophy: Selected Papers of Richard Montague, Yale University Press.
7. Pustejovsky, J. (2001), Type Construction and the Logic of Concepts, In P. Bouillon and F. Busa (eds.), The Syntax of Word Meanings, Cambridge University Press.
8. Saba, W. S. (2006), The Structure of Commonsense Knowledge, in Paolo Valore (Ed.), Topics on General and Formal Ontology, pp. 221-243, Polimetrica International Scientific Publisher.
9. Saba, W. S. (2001), Language and Commonsense Knowledge, In Brooks, M., Corbett, D., Stumptner, M., (Eds.), AI 2001: Advances in Artificial Intelligence, LNAI Vol. 2256, pp. 426-437, Springer.
10. Weiskopf, D. A. (forthcoming), Compound nominals, Context and Compositionality, to appear in Synthese.

Appendix: (Non-Sense → Logical Contradiction)

Using the logical formulation of the meaning of former given above, we show here how the concept 'former father' translates into a logical contradiction.

First, we reiterate the meaning of 'former' in (1). In (2) we state the fact that the role type Father has an essential temporal property, namely that once someone is a father they are always a father. The deductions that follow should be obvious.

1. $(\forall x : \text{Role})\big(\text{former}(x) \equiv_{df} \lambda P\big[(\exists t)\big((t < \text{now}) \wedge P(x,t) \wedge \neg P(x,\text{now})\big)\big]\big)$
2. $(\forall x)\big((\exists t_1)\big(\text{fat her}(x,t_1) \supset (\forall t_2)\big((t_2 > t_1) \supset \text{fat her}(x,t_2)\big)\big)\big)$
3. $(\exists t)\big((t < \text{now}) \wedge \text{fat her}(x,t) \wedge \neg \text{fat her}(x,\text{now})\big)$ (1) applied on father
4. $(t < \text{now}) \wedge \text{fat her}(x,t) \wedge \neg \text{fat her}(x,\text{now})$ EI of (3)
5. $\text{fat her}(x,t)$ $\wedge$ – elimination of (4)
6. $(\exists t_1)\big(\text{fat her}(x,t_1) \supset (\forall t_2)\big((t_2 > t_1) \supset \text{fat her}(x,t_2)\big)\big)$ UG of (2)
7. $\text{fat her}(x,u) \supset (\forall t_2)\big((t_2 \geq u) \supset \text{fat her}(x,t_2)\big)$ EI of (6)
8. $(\forall t_2)\big((t_2 > t) \supset \text{fat her}(x,t_2)\big)$ (5), (7) and MP
9. $(t_2 > t) \supset \text{fat her}(x,t_2)$ UG of (8)
10. $(t < \text{now})$ $\wedge$ – elimination of (4)
11. $\text{fat her}(x,\text{now})$ (9), (10) and MP
12. $\neg \text{fat her}(x,\text{now})$ $\wedge$ – elimination of (4)
13. $\perp$ (11) and (12)

Speech Processing

Acoustic Model Adaptation for Codec Speech based on Learning-by-Doing Concept

Shingo Kuroiwa, Satoru Tsuge, Koji Tanaka, Kazuma Hara, and Fuji Ren

Faculty of Engineering, The University of Tokushima,
Tokushimashi 770-8506, Japan
{kuroiwa,tsuge,ren}@is.tokushima-u.ac.jp,
WWW home page: http://ai-www.is.tokushima-u.ac.jp/

Abstract. Recently, personal digital assistants like cellular phones are shifting to IP terminals. The encoding-decoding process utilized for transmitting over IP networks deteriorates the quality of speech data. This deterioration causes degradation in speech recognition performance. Acoustic model adaptations can improve recognition performance. However, the conventional adaptation methods usually require a large amount of adaptation data. In this paper, we propse a novel acoustic model adaptation technique that generates "speaker-independent" HMM for the target environment based on the learning-by-doing concept. The proposed method uses HMM-based speech synthesis to generate adaptation data from the acoustic model of HMM-based speech recognizer, and consequently does not require any speech data for adaptation. By using the generated data after coding, the acoustic model is adapted to codec speech. Experimental results on G.723.1 codec speech recognition show that the proposed method improves speech recognition performance. A relative word error rate reduction of approximately 12% was observed.
Keywords: Speech Recognition, Model Adaptation, Codec Speech, Speech Synthesis, Learning-by-Doing

1 Introduction

In recent years, telephone speech recognition systems encompassing thousands of vocabularies have become practical and widely used [1, 2]. These systems are generally utilized by automatic telephone services for booking an airline ticket, inquiring about stock, receiving traffic information, and so on. However, the recognition accuracy of cellular phones is still inadequate due to compression coding or ambient noise [3–5].

Recently, personal digital assistants like cellular phones are shifting to IP terminals. For transmission over IP networks, speech data must be encoded at the sending end and subsequently decoded at the receiving end. This coding process deteriorates the quality of the voice data. Although most people can not notice this deterioration, it seriously affects the performance of those speech recognizers not designed for low-quality voice data[6]. The major causes of speech recognition performance degradation are : distortion in the transmission environment (transmission error and packet loss), and low bitrate speech coding (loss

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 105-114

of speech information). These distortions cause a mismatch between the feature vectors of input speech and acoustic models[7, 4].

The best way to overcome this degradation is by collecting a large amount of data in the target enviroment and training acoustic models using them. However, this method requires huge costs. Adaptation methods, such as MLLR (Maximum Likelihood Linear Regression) [8] or MAP (Maximum A Posterior probability)[9] also require a large amount of adaptation data to estimate "speaker-independent" models[3]. Actually, in [3] they used at least 1,000 utterances from 30 speakers to estimate codec-dependent HMMs.

We propose in this paper novel adaptation methods based on a learning-by-doing concept, in which a speech recognition system utters sentences in a target environment and adapts acoustic models by listening to them. This method does not need codec speech data for adaptation because these data are generated by speech synthesis from the acoustic model. By using the generated data after coding, the acoustic model is adapted to codec speech. Consequently, this method can adapt the acoustic model to various codec speech without any speech data if the coding method is specified.

Presented in section 2 is the effect of the speech coder for use with IP telephones on speech recognition. Section 3 presents our approach and section 4 presents our experiments, followed by conclusions and an outline for future work.

2 Infuluence of the speech codec on speech recognition

2.1 Baseline method

In codec speech recognition, the easiest and ideal method is to use an acoustic model that is trained with codec speech. A diagram of this training method is shown in Figure 1. This method requires a large quantity of codec speech for training an accurate acoustic model.

In order to verify the effect of the speech coder on speech recognition, we evaluated recognition performance using an acoustic model which was trained with codec speech.

For the speech coder, we selected the G.723.1 Annex A (6.3 and 5.3 kbps)[10] which has the lowest bitrate in the ITU-T H.323 recommendation[11].

2.2 ITU-T G.723.1 speech codec

The G.723.1 standard[10] is an analysis-by-synthesis linear predictive coder and it provides a dual coding rate at 6.3 and 5.3 kbps. For the higher rate of 6.3kbps, the encoder uses multipulse maximum likelihood quantization (MP-MLQ). For the lower late of 5.3kbps, the encoder employs an algebraic code excited linear prediction (ACELP) scheme. An option for variable rate operation is available using voice activity detection (VAD), which compresses the silent portions.

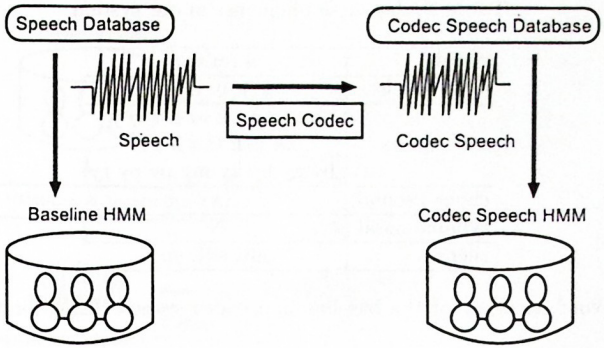


Fig. 1. A diagram of training method

Table 1. Acoustic analysis conditions

sampling rate	8 kHz
window	Hamming
FFT point	256
frame length	25 ms
frame shift	10 ms
feature vector	0-12 mel-cepstral coefficients[13](CMS) + delta + delta-delta (total 39)

2.3 Experimental conditions

The baseline acoustic models (*baseline model*) were trained with ASJ speech databases of phonetically balanced sentences[15]. Training data consist of 5,168 utterances (sampled 8kHz and 16bit) from 103 male speakers. The codec speech acoustic models (*codec speech HMM*) were trained with the same training data but they were coded by G.723.1 Annex A (6.3kbps, 5.3kbps). For the open test set, 100 utterances (1,578 words) from 23 male speakers, which are the utterences of Japanese standard dictation task in Japanese newspaper article sentence speech corpus (JNAS)[15], were used. The acoustic analysis conditions are shown in Table 1. The speech signals were windowed by a 25ms Hamming window with a 10ms shift, the mel-cepstral coefficients were obtained by mel-cepstral analysis[12, 13]. The 39-dimensional feature vector was comprised of 13 mel-cepstral coefficients (0-12th with CMS) including their delta and delta-delta coefficients. We utilized a SPTK[13] for acoustic analysis and HTK[14] for HMM training.

Table 2 shows Japanese phoneme set in our system. The "silB" and "silE" denote the silence at the beginning/end of the speech. For the acoustic model, shared state triphone HMMs with sixteen Gaussian mixture components per state were trained. The number of states was approximately 1,000. We used

Table 2. Japanese phonemes in our system

vowels	a i u e o
long vowels	a: i: u: e: o:
consonants	b d g p t k m n r w y ch j sh ts f h s z by gy hy ky my ny py ry
choked sound	q
syllabic nasal	N
silence	silB silE sp

Table 3. Word accuracy of the baseline and codec speech HMM for G.723.1 codec speech

Acoustic Model	bitrate of codec speech	
	6.3kbps	5.3kbps
baseline model	80.4 %	76.1 %
codec speech HMM	83.0 %	80.7 %

a Julius[16, 17] for the recognizer with a 20,000-word lexicon and the *tri*-gram language model.

The recognition performance is evaluated by word accuracy using the follwing equation :

$$Accuracy = \frac{N - D - S - I}{N} \cdot 100\ (\%), \tag{1}$$

where N is the total number of words, D is the number of deletions, S is the number of substitutions, and I is the number of insertions.

2.4 Evaluation of codec speech model

Table 3 shows evaluation results of the *baseline model* and *codec speech HMM*. From the figure, the word accuracy of the codec speech is lower than that of uncoded speech. Also, *codec speech HMM* achieved higher performance than that of uncoded speech HMM (*baseline model*).

These results indicate that HMM trained with codec speech data can improve the recognition performance for codec speech. However, it is difficult to obtain a large quantity of codec speech data for every coding method. Therefore, HMM adaptation methods that do not require large quantities of codec speech data are desired.

3 Adaptation using synthetic speech

A problem of conventional HMM adaptation methods is that they require a large quantity of training data for adaptation. The proposed method generates adaptation data from the HMM using a HMM-based speech synthesizer. Figure 2 shows

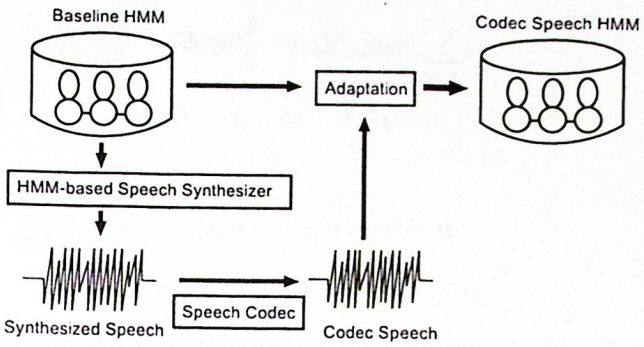


Fig. 2. Adaptaion method using speech synthesis based on HMM

a diagram of the proposed method. This method consists of HMM-based speech synthesis and HMM adaptation by the codec synthetic speech. The proposed adaptation process is : First, the HMM-based speech synthesizer generates (1) 503 phonetically-ballanced sentences[18], or (2) speech waveforms corresponding to all output distributions of all states. Next, a speech coder encodes and decodes these waveforms. Finally, baseline HMM is adapted using the encode-and-decode waveforms. This method does not require speech data for adaptation and it is applicable to any coder if the input and output of a waveform is known.

3.1 HMM-based speech synthesis

In this section, we describe the the HMM-based speech parameter generation algorithm according to [12].

Let $\boldsymbol{q} = \{q_1, q_2, \cdots, q_T\}$ be the state sequence and $\boldsymbol{o} = [\boldsymbol{o}'_1, \boldsymbol{o}'_2, \cdots, \boldsymbol{o}'_T]'$ be the vector of the output parameter sequence generated along with a single path $\boldsymbol{q}$ in the same manner as the Viterbi algorithm. The output distribution of each state is assumed to be a single Gaussian distribution for convenience of explanation.

For a given continuous HMM λ, the output speech parameter sequence $\boldsymbol{o}$ is obtained by maximizing $P(\boldsymbol{q}, \boldsymbol{o}|\lambda, T)$ with respect to $\boldsymbol{q}$ and $\boldsymbol{o}$. Since all HMMs used in the system were left-to-right models with no skipping, the probability of state sequence $\boldsymbol{q}$ is determined only by explicit state duration densities $p_q(d_q)$, i.e., the probability of d_q consecutive observations in state q. This HMM λ is the baseline HMM in Figure 2. Let a_d be a scaling factor on state duration scores and Const. be the normalization factor of Gaussian distributions, then the logarithm of $P(\boldsymbol{q}, \boldsymbol{o}|\lambda, T)$ can be written as :

$$\begin{aligned} &\log P(\boldsymbol{q}, \boldsymbol{o}|\lambda, T) \\ &= a_d \log(\boldsymbol{q}|\lambda, T) + \log P(\boldsymbol{o}|\boldsymbol{q}, \lambda, T) \end{aligned}$$

$$= a_d \sum_{k=1}^{K} \log p_{qk}(d_{qk}) - \frac{1}{2}\log|\Sigma| - \frac{1}{2}(o-\mu)'\Sigma^{-1}(o-\mu) - \text{Const.} \quad (2)$$

where

$$\mu = [\mu_{q1}, \mu_{q2}, \cdots, \mu_{qT}] \quad (3)$$

$$\Sigma = \text{diag}[\Sigma_{q1}, \Sigma_{q2}, \cdots, \Sigma_{qT}] \quad (4)$$

and μ_{qt} and Σ_{qt} are the mean vector and the covariance matrix associated with state q_t, respectively. We assume that the total number of states which have been visited during T frames is K ($\sum_{k=1}^{K} d_{qk} = T$).

By using a MLSA (Mel Log Spectral Approximation) filter[19] speech is synthesized from the generated sequence o.

3.2 Acoustic model adaptation using synthesized 503 sentences

The first adaptation method is simple. First, the HMM-based speech synthesizer generates 503 phonetically-ballanced sentences[18]. We consider that the synthsized speech was uttered by one "speaker-independent" speaker because it is synthesized from a "speaker-independent" HMM. Next, a speech coder encodes and decodes these utterances. Finally, a codec-speech HMM is estimated with MLLR using these codec utterances. One full matrix for a global regression class is used as a transformation matrix of MLLR[8].

3.3 Acoustic model adaptation using waveforms corresponding to mean vectors

The second proposed method uses synthetic speech segments corresponding to the mean vector of each output distribution.

The output distribution of state i is defined as follows:

$$b_i(o) = \sum_{k=1}^{M} c_{im}\mathcal{N}(o, \mu_{im}, \Sigma_{im}) \quad 1 \le i \le N \quad (5)$$

$$\mathcal{N}(o, \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^n|\Sigma|}}\exp\left(-\frac{1}{2}(o-\mu)'\Sigma^{-1}(o-\mu)\right) \quad (6)$$

where M is the number of Gaussian mixtures, N is the number of states, μ_{im} the mean vector and Σ_{im} is the covariance for the output probability functions of mixture m at state i. c_{im} is the mixture weight of mixture m at state i.

The following process is performed on all output distributions:

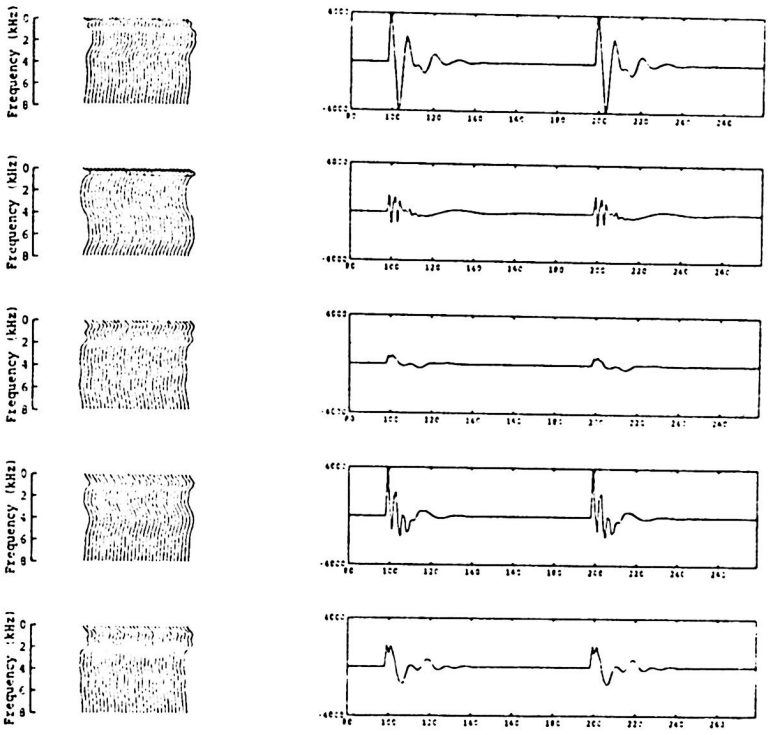


Fig. 3. Japanese vowel (/a/,/i/,/u/,/e/,/o/) spectrums and waveforms generated from the mel-cepstral coefficients

1. Generate L frame speech from the mean vector μ_{im}, using the speech synthesis algorithm described in section 3.1.
2. Encode and decode the synthetic speech by a speech coder and decoder.
3. Analyze the mel-cepstral coefficients $(c'_1, c'_2, \cdots, c'_L)$ from the codec speech.
4. Replace the mean mel-cepstral coefficients,

$$\bar{c}'_L = \frac{\sum_{l=1}^{L} c'_l}{L}, \tag{7}$$

with the mean vector μ_{im} of the HMM. The delta and delta-delta parameters are not adapted in this paper.

Figure 3 shows an example of the Japanese vowel (/a/,/i/,/u/,/e/,/o/) spectrums and waveforms generated from the mel-cepstral coefficients of the HMM mean vectors.

4 Experiments

To evaluate the proposed methods, we compared the recognition performance of the following HMMs :

1. the HMM trained by uncoded speech (*baseline model*),
2. the HMM adapted using 503 synthetic speech described in section 3.2 (*503 sentences*),
3. the mean-vector adapted HMM described in section 3.3 (*mean-vector based*),
4. the HMM trained with codec speech training data (*codec speech HMM*).

We consider that *codec speech HMM* shows the upper limit of the proposed method.

4.1 Conditions

Acoustic analysis and HMM training conditions are the same as detailed in section 2.3. The feature vector is comprised of 13 mel-cepstral coefficients (0-12th), and their delta and delta-delta coefficients. For the acoustic model, shared state triphone HMMs with sixteen Gaussian mixture components per state were trained.

For the *mean-vector based* adaptation method, the adaptation data corresponding to each distribution of each state, were generated at a 150 Hz pitch frequency by the MLSA filter. From a preliminary experiment, the adaptation data for unvoiced sounds were also excited with a 150 Hz pitch. For this experiment the length of each data was 0.3 seconds.

4.2 Experimental results

The experimental results are provided in Table 4. As shown by this table, the *mean-vector based* adaptation method improves the word accuracy of the *baseline model*. The proposed method was effective in both G.723.1 Annex A coders of 6.3kbps and 5.3kbps. We observed an improvement in word accuracy of approximately 1.5 points (8% relative error reduction) at 6.3kbps and about 3 points (12% relative error reduction) at 5.3kbps. The proposed method slightly degrades the recognition performance of the *codec speech HMM*. However, the proposed method did improve the recognition performance of the *baseline* without any training speech.

On the other hand, the HMM adapted using 503 synthetic speech did not improve the accuracy. One reson for this discrepancy may be that we did not study the structure of MLLR transformation matrix sufficiently. We also expect an essential problem is that the synthetic speech was one average speaker's voice.

Table 4. Word accuracy of proposed method for G.723.1 codec speech

HMM or adaptation method	bitrate of codec speech 6.3kbps	5.3kbps
baseline model	80.4 %	76.1 %
503 sentences	76.3 %	73.5 %
mean-vector based	**82.0** %	**79.0** %
codec speech HMM	83.0 %	80.7 %

5 Summary

In this paper, we propose novel acoustic model adaptation methods based on a learning-by-doing concept, in which a speech recognition system utters sentences in a target environment and adapts acoustic models by listening to them. The proposed methods generate adaptation data from the acoustic models (HMMs) by using HMM-based speech synthesis. By using the generated data after coding, the system adapt the acoustic models using these data. Experimental results of G.723.1 codec speech recognition indicated that the proposed mean-vector based adaptation method improved the recognition accuracy of codec speech when compared to the non-adaptation HMM. Additionally the word accuracy of the proposed method approaches that of the codec speech HMM.

For this study, only the MFCC mean vectors of HMM were adapted. Currently we are trying to adapt the covariance matrix. In the future, the proposed method will be adapted to other coding and environments.

Acknowledgments

This research has been partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Young Scientists (B), 15700163, Grant-in-Aid for Exploratory Research, 17656128, 2005, Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research (B), 14380166, 17300065, 2005, and International Communications Foundation (ICF).

References

1. Dahl, D. (ed.): Pracrical Spoken Dialog Systems, Kluwer Academic Publishers, Dordecht (2004)
2. Cohen, M., Giangola, J., Balogh, J.: Voice User Interface Design, Addison Wesley, Boston (2004)
3. Naito, M., Kuroiwa, S., Kato, T., Shimizu, T. and Higuchi, N.: "Rapid CODEC Adaptation for Cellular Phone Speech Recognition" Proc. European Conf. Speech Communication and Technology (2001) 857–860
4. TanakaK., Kuroiwa, S., Tsuge, S. and Ren, F.: "The Influence of Speech Coders for IP Telephone on Speech Recognition" Proc. Int. Conf. Information-2002, **3**, (2002) 44–48

5. Kato, T., Naito, M., and Shimizu, T.: "Noise-Robust Cellular Phone Speech Recognition Using Codec-Adapted Speech and Noise Models", IEEE Int. Conf. Acoustics, Speech, and Signal Processing (2002) 1315–1318
6. LillyB.T. and PaliwalK.K.: "Effect of Speech Coders on Speech Recognition Performance", Proc. Int. Conf. Spoken Language Processing (1996), 877–880
7. Gallardo-Antolin, A., Pelaez-Moreno, C., and Diaz-de-Maria, F.: "A robust front-end for ASR over IP and GSM networks: an integrated sinario" Proc. European Conf. Speech Communication and Technology (2001) 1691–1694
8. Leggetter, C.J. and Woodland, P.C.: "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models" Computer Speech and Language, **9** (1995), 171–185
9. Gauvian, J.L. and Lee, C.H.: "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains" IEEE Trans. Speech and Audio Processing, **2** (1994) 291–298
10. ITU-T Recommendations: "ITU-T G.723.1 Annex A – Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s," International Telecommunication Union (1996)
11. ITU-T Recommendations: "ITU-T H.323 – Packet-based multimedia communications systems," International Telecommunication Union (2003)
12. Tokuda, K., Yoshimura, T., Masuko,T., Kobayashi, T. and Kitamura, T.: "Speech parameter generation algorithms for HMM-based speech synthesis," IEEE Int. Conf. Acoustics, Speech, and Signal Processing (2000) 1315–1318
13. "Speech Signal Processing Toolkit" http://kt-lab.ics.nitech.ac.jp/tokuda/SPTK/
14. "HTK Hidden Markov Model Toolkit" http://htk.eng.cam.ac.uk/
15. Ito, K., Yamamoto, M., Takeda, K., Takezawa, T., Matsuoka, T., Kobayashi, T., Shikano, K., and Itahashi, S.: "JNAS:Japanese speech corpus for large vocabulary continuous speech recognition reseach," Journal of the Acoustical Society of Japan E, **20** (1999) 199–207
16. Lee, A., Kawahara, T., and Shikano, K.: "Julius — an open source real-time large vocabulary recognition engine," Proc. European Conf. Speech Communication and Technology (2001) 1691–1694
17. "Julius – Open-Source Large Vocabulary CSR Engine," http://julius.sourceforge.jp/en/julius.html
18. Kurematsu, A., Takeda, K., Sagisaka, Y., Katagiri, S., Kuwabara, H. and Shikano, K.: "ATR Japanese Speech Database as a Tool of Speech Recognition and Synthesis", Speech Communication, **9** (1990) 357–363
19. Imai, H.: "Cepstral analysis synthesis on the mel frequency scale" IEEE Int. Conf. Acoustics, Speech, and Signal Processing (1983) 93–96

Specific Speaker's Japanese Speech Corpus over Long and Short Time Periods

Satoru Tsuge[1], Masami Shishibori[1], Fuji Ren[1],
Kenji Kita[2], and Shingo Kuroiwa[1]

[1] Faculty of Engineering, the University of Tokushima,
2-1, Minami Josanjima, Tokushima, Japan
tsuge@is.tokushima-u.ac.jp

[2] Center for Advanced Information Technology, the University of Tokushima,
2-1, Minami Josanjima, Tokushima, Japan

Abstract. It is known that speech recognition performance varies pending when the utterance was uttered although speakers use a speaker-dependent speech recognition system. This implies that the speech varies even if a specific speaker utters a specific sentence. Hence, we investigate the speech variability of a specific speaker over short and long time periods for getting the stable speech recognition performances. For this investigation, we need a specific speaker's speech corpus which is recorded over long time periods. However, at present, we have not seen such a Japanese speech corpus. Therefore, we have been collecting the Japanese speech corpus for investigating the relationship between intra-speaker speech variability and speech recognition performance. In this paper, first, we introduce our speech corpus. Our corpus consists of six speakers' speech data. Each speaker read specific utterance sets tree times a day, once a week. Using a specific female speaker's speech data in this corpus, we conduct speech recognition experiments for investigating the relationship between intra-speaker speech variability and speech recognition performance. Experimental results show that the variability of recognition performance over different days is larger than variability of recognition performance within a day.

1 Introduction

Recently, speech recognition systems, such as car navigation systems, and cellular phone systems have come into wide use. Although a speaker uses a speaker-dependent speech recognition system, it is known that speech recognition performance varies pending when the utterance was uttered. For this reason, we consider that speech characteristics varies even though the speaker and utterance remain constant. This intra-speaker variability is caused by some factors including emotion and background noise. If the recognition performance is not consistent, then products using speech recognition systems become less useful for the end-user. As the relationship between intra-speaker's speech variability and speech recognition performance is yet unclear, we began to investigate the nature of this relationship.

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 115-124

In the field of speaker identification and verification, it has been reported that speaker verification performance degraded for a standard set of templates after only a few months[1][2]. However, we have not seen this evaluation applied to Japanese speech recognition. At present, there are a lot of Japanese speech corpora for studying speech recognition[3][4][5]. However, we have not seen a corpus of Japanese speech data of a specific speaker over a long time period. Hence, we have not been able to investigate the relationships between the intra-speaker's speech variability and speech recognition performance. In order to examine the intra-speaker's speech variability and its influence on speech recognition performance, we need a new corpus. Consequently, we started collecting some specific speaker's read speech data. Data collection was initiated in October 2002. It is still underway as of December 2005. In this paper, we describe the speech corpus collected by us for investigating intra-speaker's speech variability.

Because the initial speech data which were collected at one recording time was one file, we need to divide one recorded file into separate utterances. This process requires a lot of time and effort. In this paper, we propose an automatic utterance segmentation tool for dividing one recored file into separate utterances.

At present, we have some speech data which has been processed using this segmentation tool. We conducted the speech recognition experiments using these speech data. In this paper, we report phoneme accuracy on each speaking day and at each speaking time.

In the following section, we introduce the our database. In section 3, we propose the automatic utterance segment tool for processing our database. In section 4, we show the experimental conditions and results. In the last section (section 5), we describe the summary of this paper and future works.

2 A Japanese speech corpora

There are a lot of Japanese speech corpora for studying speech recognition[3][4][5]. Most of these speech corpora are designed for studying speaker independent speech recognition systems. Hence, the amount of speech data from one speaker is limited and often collected on one day. Using these speech corpora, it is difficult to investigate the speaker's speech variability over long time periods and relationships between speaker's speech variability and speech recognition performance. In addition, these corpora lack information about speakers, such as physical condition of speaker, environmental condition of recording, and so on. To investigate what caused the variability of the speaker's speech, we need this information. Consequently, we began collecting speech data of some specific speakers uttered over a long time period. In our corpus, the speaker fills out a questionnaire which is described in section 2.5 at each recording session. Since September 2002, we have been collecting speech data for investigating the relationships between the speech variation and speech recognition performance. In this section, we describe our Japanese speech corpus.

2.1 Speakers and recording days

Our corpus consists of six speakers' speech data. The number of male speakers and the number of female speakers are four and two, respectively. Each speaker read utterance sets, described in section 2.3, three times a day, once a week. The length of each recording was about fifteen minutes.

2.2 Recording environments

Our corpus was collected in one of the two following types of recording environments,

- Schoolroom
  We used a quiet school room for recording from November 2002 to October 2003.
- Silent room
  We used a silent room for recording from October 2003 to the present.

2.3 Utterance sets

We used the two utterance list sets for recording. In this paper, we call these Common recording set and Individual recording set. The contents of each are described below:

- Common recording set
  - Japanese phonetically balanced sentences (The number of sentences is 50. These sentences are called Aset.)
  - Isolated words (The number of words is 10.)
  - Name words (The number of words is 10.)
  - 4 digit strings (The number of items is 10.)
  - Checked sentences (The number of sentences is 16.)
- Individual recording set
  - Japanese phonetically balanced sentences
  - Isolated words
  - 4 digit strings
  - Japanese newspaper sentences

  The items in the individual and common recoding set differ.

All speakers uttered the common recording set at every recording session. The length of this recording set, which included non-voice sections and mistaken sections, is about thirteen minutes. The contents of the individual recording set were different at each recording session.

2.4 Recording file format

We used the head set microphone, Sennheiser HMD410, and the DAT recorder, Sony TCD-D100, for the recording system. The DAT's sampling rate is 48 kHz. Using DAT link, we copied the recorded speech data from DAT to the computer. At that time, the number of recorded speech data files was one because this data included the non-voice sections and the mistaken sections. Then, we divided this speech data file to individual speech data. Finally, we resampled the speech data at 16kHz.

2.5 Questionnaire

For investigating the reason of intra-speaker's speech variability, the speaker filled out a questionnaire at every recording session. The contents of the questionnaire are listed below:

- Physical conditions
  - Body temperature
  - Weight
  - Percentage of body fat
  - Pulse rate
  - Blood pressure
  - Feeling or Mood
  - Condition of nose and throat
- Environmental conditions
  - Outdoor temperature
  - Outdoor humidity
  - Temperature in recording room
  - Humidity in recording room
  - Day of recording
  - Time of recording

In addition, after the last recording session in a day, the speaker answers some questions about today's activities and the hours of sleep yesterday.

3 Automatic utterance segment tool

The speaker has to check each utterance each time if we collect the utterances individually. In general, the speech is recorded as one file from recording start to recording end. Hence, there are some noises, non-voiced sections and mistaken sections in this file. For our speech corpus, we have to segment this file into individual utterances and select the useful utterances. However, this process requires a lot of time and effort. In this paper, we propose an automatic utterance segmentation tool for dividing the recorded speech files.

Figure 1 illustrates the flow of the proposed automatic utterance segmentation tool. Below, we explain each part of the proposed method.

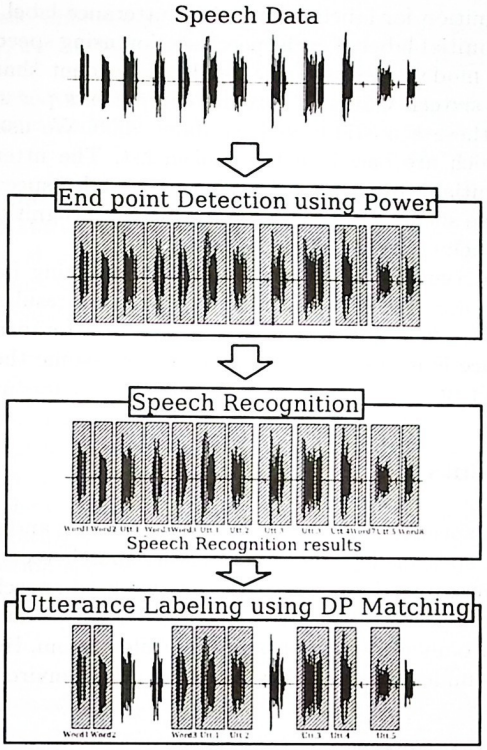


Fig. 1. The flow of the automatic utterance segment tool.

1. End point detection using power
   First, we find the end point of each utterance in the speech data using power. The power of m-th frame ($P(m)$) is calculated as follows,

$$P(m) = \frac{\sum_{n=0}^{T-1}(s((m*S)+n))^2}{T}, \tag{1}$$

   where T, S, and $s(n)$ are the the frame length, the frame shift, and speech signal, respectively. Using this, we detect the end point. In this paper, the frame length and frame shift are set at 5 msec and 2 msec, respectively. If the power of a frame following a silence is bigger than the threshold, this frame is the start of the speech section. Using the threshold selection method[6], we determine the threshold for detecting the end point. After the start section, we find the next silint segment using power then we define the end point of the speech section. However, we ignore sections shorter than 300msec. We added a 1 second silence to speech sections.

2. Speech recognition for labeling the initial utterance label
   We made on initial label of each speech section using speech recognition. For the acoustic models, we used the gender dependent shared-state triphone HMMs with sixteen Gaussian mixture components per state. The number of states of these acoustic models is about 2000. We use a dictionary and grammar which are based on the spoken list. The utterance sections are labeled the initial utterance labels which are included speech list. In addition, we modify the silent period to 0.5 second using recognition result.
3. Utterance labeling using DP matching
   After speech recognition, we conducted DP matching between the spoken list, which is a corrected list, and the recognition results. This process removed the mistaken sections from the separate utterances. In this process, if an utterance is uttered more than once, we assume the last utterance to be the correct utterance.

4 Experiments

For investigating speech recognition variability over long and short time periods, we conducted a speaker-dependent continuous speech recognition experiment using our speech corpus. In this experiment, we used speech data which were downsampled from 48kHz to 16kHz. The collected speech data were recorded in two types of environments: a schoolroom and a silent room. Hence, we conducted two experiments under the condition of the recording environments.

4.1 Experimental Conditions

Training data 502 Japanese phonetically balanced sentences were used for the training for schoolroom recording environment and 503 Japanese phonetically balanced sentences were used for training for the silent room recording environment, respectively. These training sentences were uttered on following days:

- Schoolroom recording environment
  2002.11.12, 19, 26, 2002.12.3, 10, 17, 24, 2003.1.14, 21
- Silent room recording environment
  2004.11.9

Testing data For the testing data, we used 50 kinds of Japanese phonetically balanced sentences. These sentences were uttered three times in a each recording day. These sentences were recorded on the following days:

- Schoolroom recording environment
  2002.11.19, 26, 2002.12.3, 10, 17, 24, 31, 2003.1.7, 14, 21, 28, 2003.2.4, 11, 18, 25, 2003.3.4, 12, 18, 26, 2003.4.2, 7, 14, 21, 28, 2003.5.6, 12, 19, 26, 2003.6.3, 10, 17, 24, 2003.7.1, 8, 15, 22, 2003.8.5, 11, 17, 30, 2003.9.1, 10, 16, 23, 2003.10.03

- Silent room recording environment
  2003.10.10, 17, 24, 31, 200311.07, 15, 21, 28, 2004.1.5, 9, 16, 23, 30, 2004.2.6.

For the testing set, we used 6,747 and 2,099 utterances in the schoolroom recorded environment and the silent room recorded environment, respectively[3].

Feature vector and acoustic model The feature vector for the experiment was 25 MFCCs (12 static MFCCs + 12 of their deltas + one delta-logpower). For the acoustic model, shared-state triphone HMMs with sixteen Gaussian mixture components per state were trained. We set the number of states at about 270. We used the same conditions in both recording environments.

Decoder and evaluation For the decoder, we used the one-pass Viterbi algorithm with the phonotactic constraints of Japanese language expressed. Recognition results are given as phoneme accuracy. We use HTK version 3.2.1[7] as acoustic modeling and recognition tools.

We calculated the variance of recognition accuracy for investigating the variability. For investigating the influence of speaking time, we calculated the variance of the recognition accuracy in equation (2).

$$V_t = \frac{\sum_{d \in date} (ACC_{d,t} - AVE_t)^2}{N_{date}}, \tag{2}$$

where, *date* indicates all speaking days, $ACC_{d,t}$ and AVE_t are the recognition accuracy of speaking day d and speaking time t and the average recognition accuracy of speaking time t. N_{date} is the number of speaking days. To investigate the influence of the speaking time, we also calculated the variance of the recognition accuracy in equation (3).

$$V = \frac{\sum_{d \in date} \sum_{t \in time} (ACC_{d,t} - AVE_d)^2}{N_{date} * N_{time}}, \tag{3}$$

where, *time* indicates the all speaking times. AVE_d indicates the average recognition accuracy of the speaking day d. N_{time} is the number of speaking times in a day.

4.2 Experimental results

In this section, we present the experimental results in order to investigate the influence of speaking days and speaking time. Tables 1 and 2 show the speech recognition performance on the schoolroom recording environment and on silent room recording environment, respectively. Table 3 shows the variances which were calculated by equation (2) and equation (3).

[3] For recording mistakes, the number of testing sentences is 49 in afternoon on 2003.1.14, morning on 2003.4.28, morning on 2003.7.1, and evening on 2003.11.28

Table 1. Schoolroom environment (phoneme accuracy (in %))

	Recording days											
	2002							2003				
	1119	1126	1203	1210	1217	1224	1231	0107	0114	0121	0128	0204
Morning	78.5	76.5	76.4	73.8	75.6	77.3	78.6	75.0	75.1	72.9	72.4	74.4
Afternoon	78.1	76.6	81.6	73.8	76.7	76.3	75.9	72.6	73.3	72.7	72.7	70.9
Evening	76.9	75.6	76.6	75.5	77.4	77.5	76.3	75.2	74.0	75.3	75.0	69.4
Average	77.9	76.2	78.2	74.4	76.6	77.0	76.9	74.3	74.1	73.6	73.4	71.6
	2003											
	0211	0218	0225	0304	0312	0318	0326	0402	0407	0414	0421	0428
Morning	69.5	67.4	71.6	73.6	67.7	71.4	72.9	74.3	74.7	73.8	75.4	71.9
Afternoon	67.9	68.4	70.2	71.2	71.1	73.4	71.7	74.1	68.7	71.2	71.5	71.4
Evening	71.4	69.6	71.5	73.4	70.7	73.0	75.1	73.6	72.4	74.5	73.5	72.8
Average	69.6	68.5	71.1	72.7	69.8	72.6	73.2	74.0	71.9	73.1	73.5	72.0
	2003											
	0506	0512	0519	0526	0603	0610	0617	0624	0701	0708	0715	0722
Morning	69.4	72.7	62.5	72.5	76.2	75.0	73.9	76.2	75.6	77.1	73.3	74.9
Afternoon	72.5	71.7	64.4	74.2	73.2	70.3	71.1	75.0	74.5	76.2	73.2	75.5
Evening	73.5	72.6	68.6	74.0	74.9	73.9	73.0	75.7	75.9	74.9	75.0	74.5
Average	71.8	72.3	65.1	73.6	74.8	73.1	72.7	75.6	75.3	76.1	73.8	75.0
	2003									Average		
	0805	0811	0817	0830	0901	0910	0916	0923	1003			
Morning	76.3	73.4	75.4	75.6	74.5	70.6	74.3	73.8	73.1	73.7		
Afternoon	74.7	74.6	75.4	71.8	73.1	72.8	74.7	71.8	70.9	73.0		
Evening	76.1	76.7	73.9	73.6	74.8	75.4	73.9	74.1	74.7	74.1		
Average	75.7	74.9	74.9	73.7	74.1	73.0	74.3	73.2	72.9	73.6		

Comparison of recognition performances

Table 1 shows that the recognition performances from 2002.11.19 to 2003.1.21 are higher than other days. These are the days when the training data was recorded. Because the recording days of the training data and the testing data were the same, we consider that there are few acoustic mismatches between the training data and the testing data. However, we can see from this table that the recognition performances on other days degraded compared to the training data recording days. I believe that the speech variability on different days was greater than the speech variability within a day.

We can see from Table 2 that the recognition performances are consistent compared to the schoolroom recording environments. We consider that the speakers became used to speaking utterances. Hence, the acoustic feature vectors vary little in testing periods.

Compared to Table 1 and 2, we can see that the phoneme accuracies of the silent room environment are lower than those of the schoolroom environment. We suppose that the reason is that the period between testing and training data in the silent room recording environment is 7 months.

On the other hand, we can see from Table 1 that the recognition performances of 2003.5.19 are lower than that of other days. Hence, we investigated the questionnaire of 2003.5.19. This questionnaire showed that the speaker caught a cold. From this result, we can confirm that the physical condition influences the recognition performance. We will investigate the detail of this result and the relationships between the emotion and recognition performance from the questionnaires.

Table 2. Silent room environment (phoneme accuracy (in %))

	Recording days										
	2003									2004	
	1010	1017	1024	1031	1107	1115	1121	1128	0105	0109	0116
Morning	72.0	70.6	68.5	68.4	71.7	67.9	70.6	70.6	71.5	70.7	72.9
Afternoon	72.2	73.4	67.3	70.8	72.0	73.5	73.2	70.6	70.8	73.3	73.6
Evening	72.1	70.2	75.1	72.9	72.3	73.2	70.7	74.6	71.4	72.0	73.7
Average	72.1	71.4	70.3	70.7	72.0	71.5	71.5	71.9	71.2	72.0	73.4

	2003			
	0123	0130	0206	Average
Morning	72.1	69.4	68.7	70.4
Afternoon	72.3	70.9	70.2	71.7
Evening	73.7	70.4	72.1	72.4
Average	72.7	70.2	70.3	71.5

Table 3. Variance of the recognition accuracy against speaking day and speaking time

recording environment	V	V_m	V_a	V_e
schoolroom	1.60	8.88	8.36	4.01
silent room	2.32	2.91	2.12	2.38

Comparison of variances

From Table 3, we also see that the difference between the variance of the speaking time in the silent room (V_m, V_a, V_e) is smaller than in the schoolroom. The training data for the silent room was collected in one day although the training data for the schoolroom was collected in two months. Hence, the acoustic model for the schoolroom may be able to be trained using the variation of the speech resulting from a longer collection period.

5 Summary

In this paper, we described a Japanese speech corpus for investigating speech variability in a specific speaker over long and short time periods. This corpus has been collected by us since October 2002. The data collection is still ongoing. We have collected six speakers' utterances. Each speaker spoke three times in a day once a week. In addition, we proposed an automatic utterance segmentation tool for dividing one recorded file to individual useful utterances.

Using a part of our speech corpus, we conducted speaker dependent speech recognition experiments. Experimental results show that recognition performance degraded when there are acoustic mismatches between testing and training data collected over different periods.

In the future, we will continue to collect speech data and enhance our speech corpus. We will conduct speech recognition experiments using other speaker's

speech data and investigate the influence of recording period on the recognition performance.

6 Acknowledgment

This research has been partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Young Scientists (B), 15700163 and the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (B), 17300065 & 17300036.

References

1. Matsui, T., Nishitani, T. , and Furui, S.: "A study of model and a priori threshold updating in speaker verification," *IEICE (D-II)*, Vol. J81-DII, No. 2, pp. 268–276, 1998, (in Japanese).
2. Hayakawa, S., Takeda, K., and Itakura, F.: "A speaker verification method which can control false acceptance rate," *IEICE (D-II)*, Vol. J82-D-II, No. 12, pp. 22212–2220, 1999, (in Japanese).
3. Ito, K., Yamamoto, M., Takeda, K., Takezawa, T., Matsuoka, T., Kobayashi, T., Shikano, K., and Itahashi, S.: "The design of the newspaper-based japanese large vocabulary continuous speech recognition corpus," *Proc. ICSLP*, pp. 3261–3264, 1998.
4. Nakamura, A., Matsunaga, S., Shimizu, T., Tonomura, M., and Sagisaka, Y.: "Japanese speech databases for robust speech recognition," in *Proc. ICSLP*, pp. 2199–2202, 1996.
5. Maekawa, K.: "Corpus of spontaneous japanese: Its design and evaluation," *Proc. of the ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition (SSPR2003)*, pp. 7–2, 2003.
6. Otsu, N.: "A threshold selection method from gray-level histograms," *IEEE Trans. Sys., Man, and Cybernetics*, Vol. SMC-9, No.1, pp. 62–66, 1979.
7. "HTK: Hidden Markov Model Toolkit," http://htk.eng.cam.ac.uk/.

A Task Analysis of Nursing Activites Using Spoken Corpora

Hiromi itoh Ozaku[1], Akinori Abe[1], Kaoru Sagara[2], Noriaki Kuwahara[1] and Kiyoshi Kogure[1]

[1] Advanced Telecommunications Research Institute International
{romi,ave,kuwahara,kogure}@atr.jp
[2] Seinan Jo Gakuin University
sagara@seinan-jo.ac.jp

Abstract. This paper illustrates our examination of the "E-nightingale Project," reports the results manually obtained from task analyses of nursing activities using spoken corpora, and describes the possibility of automated task analyses using natural language processing technologies.
Recently, medical malpractice has become a serious social problem and safety measures to prevent and decrease medical malpractice are being taken in some hospitals, for example, by building computerized database in new systems.
The Japanese Nursing Association states in its guidelines that nurses are encouraged to make nursing reports, including medical malpractice reports, and to analyze the cause of accidents, which is helpful for preventing recurrences.
However, it is very difficult for nurses to produce detailed records during their working hours except for malpractice reports. Furthermore, it is hard work for nurses on duty to analyze the recorded data in detail.
As a solution, we have launched the "E-nightingale Project," in which some systems using wearable computers are being developed for the purpose of preventing and reducing medical malpractice.
As part of this project, we built spoken corpora using voice data that were monitored and recorded daily nursing assignments in hospitals with our developed devices. 800 hours of voice data were accumulated, and 70 hours of those were transcribed as the spoken corpora. Then we started analyzing nursing tasks using the spoken corpora, and considered the possibility of automated task analysis using natural language processing technologies.

1 Introduction

Recently, medical malpractice has become a serious social problem [1]. The Japanese Ministry of Health, Labor and Welfare has reported that nursing teams are most frequently involved in medical accidents in hospitals [2]. The Japanese Nursing Association also states in its guidelines that nurses are encouraged to make nursing reports and to analyze the cause of accidents by comparing medical malpractice reports, which is helpful for preventing recurrences. Some nursing

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 125-136

tasks in particular are closely associated with the occurrence of medical malpractice. In other words, if the nursing reports and medical malpractice reports are analyzed according to nursing tasks, it can be understood why and when a medical malpractice event happened. By using the results, medical malpractice can be reduced and prevented. However, it is very difficult for nurses to make detailed records during their working hours; furthermore, it is hard work for nurses on duty to analyze the recorded data in detail.

We launched the "E-nightingale Project" to develop a nursing service support system based on multimedia data in the real field that monitors and records nursing activities in detail using wearable computers. We can obtain multimedia data from our developed devices such as video data, sensed data of body actions, and voice data. Since video data can only be obtained in a narrow range, it is not possible to analyze all of the nursing activities. In fact, we are still analyzing sensed data so we cannot yet determine nursing behaviors from the sensed data. On the other hand, nurses have to confirm task names aloud every time they start their tasks, and we can use this information to analyze their tasks. Therefore, we focus on voice data to analyze nursing activities. In addition, since we need to develop dictionaries for an automatic speech recognition, we have decided to build corpora on nursing activities. This is the first attempt to develop special devices, to create spoken corpora for understanding nursing assignments, and to analyze nursing tasks by using the spoken corpora.

We monitored nurses in a hospital for several days, and recorded multimedia data using our wearable computers. The multimedia data include video data, voice data, sensor data and so on. We tried to analyze these data automatically; however, there arose some problems with analyzing them automatically. For example regarding voice data, nurses usually use technical terms with various expressions. We have some databases that can be regarded as standard dictionaries, such as JNPSM (Japan Nursing Practice Standard Master) built by MEDIS-DC (Medical Information Center Development Center) [3], and ICNP (International Classification for Nursing Practice) [4], but since they include written language, it is difficult to match their terms and the spoken terms collected in our project. Furthermore, concrete definitions of nursing tasks in actual working places do not exist in the research area of nursing task analysis.

In this paper, we report a way to accurately monitor nurses' activities in actual working places, and propose a method to effectively analyze nursing tasks from nurses' activity data. We first illustrate our examination of developing corpora of voice data for understanding nurses' activities, then report results from task analyses of nursing activities manually, and finally explain the possibility of automated task analyses using natural language processing technologies such as text-clustering methods.

2 E-nightingale Project

As previously mentioned, preventing medical malpractice is a very important research issue. To do so, it is vital to recognise the reality of hospital situations, and

to understand nursing activities. It is also important to make accurate nursing reports.

To fully understand nursing activities, a one-day schedule of nursing tasks should be recorded and analyzed. In the traditional method, nurses have reported their activities after their work using their memories. Alternatively, one researcher has followed a nurse all day and taken meticulous notes of the nurse's activities. The traditional method, called a "Time Study Survey," is expensive and cannot fully describe their daily activities with sufficient accuracy [7], which is why we started the E-nightingale Project. The project has the following purposes:

- to develop wearable computers for easy recording of daily nursing assignments,
- to obtain multimedia data of nursing assignments in the hospital,
- to analyze the data to accurately grasp nursing tasks and to build a database of nursing tasks efficiently,
- to utilize the database by better understanding the activities and comparing malpractice reports to prevent future malpractice.

Though we can obtain multiple data types, in this paper we focus on voice data for the following reasons:

- natural language processing technologies are evolving and can be applied in the real field,
- speech recognition technologies are also evolving and can be applied in noisy fields [5, 6],
- much text information can be utilized by the diffusion of electronic charts,
- nursing activities and situations can be understood easily if the speech of nurses can be grasped.

This is to say we are developing a method and system for automatic "Time Study Survey." As the first step, we recorded the voice data of nursing activities along with cue words and cue sentences such as "I will start A task," that are regarded as annotations by nurses using our devices. In the next section, we explain how to obtain the voice data.

3 Experiments to Collect Data

We recorded voice data in some hospitals using our wearable computer. The voice data contained daily nursing assignments with many types of sounds, including talks with patients, and/or doctors, noise during nursing care, and other tasks.

First, we explain the wearable computer used in this investigation. To obtain voice data in hospitals, we used a special device comprising of an ICrecorder,

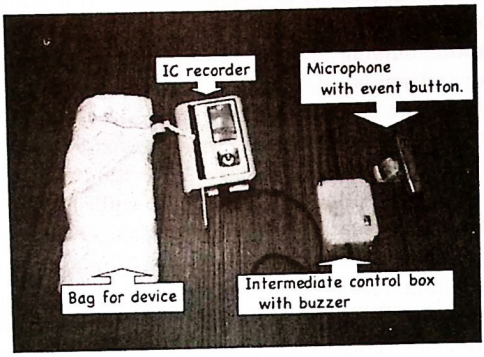


Fig. 1. Our Devices

a microphone with an event button, and an intermediate control box with a buzzer, as shown in Fig. 1.

The event button is used for explicit voice annotation when nurses start or complete a task. Recently, to decrease medical malpractice, nurses have had to confirm a medicine's name aloud when they start mixing medicines into an intravenous drip. If the voice is recorded by our device, we can correlate the voice with the nursing task "Medication Administration" manually. Furthermore, if conversations between nurses and patients can be recorded, the task performed at that time can be easily understood.

When the button is pushed, the buzzer sounds once and its sound is recorded, and then nurses record their tasks of the moment by speaking short sentences that include words of confirmation and conversations with patients. The buzzer is also set to sound periodically (every 10 minutes) to prompt nurses to make voice inputs about their ongoing tasks. Simple signal processing can extract and classify task-driven and periodic voice records as well as nurse call rings.

In two departments of a hospital, we conducted experiments on collecting data of nursing tasks through voice annotation. The nurses worked in three shifts, assigned to the primary patients of each ward. All nurses were given instructions on how to use our devices. The entire recording time was about 800 hours. Data were gathered from 163 trials for a 14-day experiment involving 39 nurses using our devices.

Next, we transcribed the voice data to text to build nursing spoken corpora — the Nursing Task Corpus and the Nursing Dialogue Corpus. The following section will include a detailed explanation of the corpora.

4 Nursing Spoken Corpus

We transcribed the voice data to text to build the nursing spoken corpora that are used for understanding nurses' tasks and collecting nursing technical terms in the daily nurse assignments, especially in clinical meetings.

The process of creating the corpora is as follows:

1. Data of the sounds, including buzzer and short sentences, are broached by the signal processing.
2. Transcriptions of each broached data are made manually.
3. Task names in a job category list are manually attached as tags to each transcription of short sentences for the Nursing Task Corpora.
4. For the Nursing Dialogue Corpora, words not included in the IPA dictionary [8] but included in voice data in conferences or clinical meetings are transcribed and tagged as "unknown words."

The transcription was made by four staff members, including an experienced specialist in making transcriptions, a nurse who had worked in hospitals for three years or longer, a pharmacist, and a part-time employee, and it was assumed that the latter three had no experience working in the field of transcription.

Two types of corpora were built from the transcribed data. One was the Nursing Task Corpus, the other the Nursing Dialogue Corpus. The Nursing Task Corpus includes transcriptions of voice data for 10 seconds with a buzzer that were extracted by simple signal processing from the voice data of a one-day assignment. The Nursing Task Corpus includes all activities by a nurse in one day. This is useful for making accurate nursing reports. If the Nursing Task Corpus is analyzed together with medical malpractice reports in detail, nurses can find when malpractice events happen, what nurses do, and how such events occur and so on. The Nursing Dialogue Corpus includes all conversations during clinical meetings. We can thus easily understand the communications among nurses, and collect technical terms in the real field for automatic speech recognition from the Nursing Dialogue Corpus.

In the next section, we briefly explain the Nursing Dialogue and Nursing Task Corpora.

4.1 Nursing Dialogue Corpus

During nurses' shift changes, they hold clinical meetings to discuss patient information, and in the process, they modify and confirm this information. Furthermore, if problems occur in their work area, they hold brief conferences to solve them by discussing their experiences.

We think dialogue in the clinical meetings includes cue words of nursing activities, their work flows, nursing technical words, and so on, and consider the cue words to be very useful for understanding their tasks. In addition, it is also useful as a language model to transcribe the voice data automatically by using speech recognition tools. We collected technical terms from dialogues in clinical meetings, especially medicine names, and medical test names. Because these names are treated as "unknown words," they do not exist in commonly used dictionaries. An example from the corpus is shown in Table 1. In this case,

"ENTO"[3] is an unknown word and should be treated as a cue word. Because "ENTO" helps to confirm nursing activities [9].

Table 1. Example from the Nursing Dialogue Corpus

Time	Nurse-ID	Utterance
00:50:19	A	Kogure-san will be discharged tomorrow. Please do not forget the ENTO sheet.
00:50:20	B	Yeah, .. ENTO sheet..
00:50:40	C	The drip infusion is still being given, isn't it?
00:50:42	A	Yes. It's still dripping now.
00:50:44	B	Ah, until when?
00:50:45	A	Until tomorrow. Until 6 o'clock.

Supporting communication between doctors and patients is vital when doctors give informed consent. As such, some researchers have implemented special communication support systems tailored for communication between doctors and patients [10]. We think our corpora are helpful for clarifying communication mechanism in hospitals.

However, the nurses' dialogues have not only a complex communication mechanism, but also many expressions such as nursing technical terms. For instance, in general "せんがん (sengan)" means "洗顔 (washing face)," but if it is narrated before surgery in ophthalmology, it means "洗眼 (washing eye)." "自立する (jiritsu-suru)" means "standing walk" if it is narrated in a conversation among nurses, but it means "to earn one's living by oneself" in general, as when it is narrated in a conversation between a nurse and a patient's family. Thus some words have multiple meanings according to the situation. As a result, sometimes nurses, especially novice nurses, experience misunderstanding in their communications. Another example is "ほりゅう (horyu)," which means "suspend giving medicine" or "keep a sting stung." Therefore, we think it will be possible to gather many words that have multiple meanings when we check the corpus. This will allow us to build or extend the contents of a dictionary of multiple-meaning expressions. We consider these expressions to include cue words that are nursing technical terms, and as such they are useful as language models to automatically recognize speech and to prevent malpractices due to misunderstanding.

4.2 Nursing Task Corpus

For the Nursing Task Corpus, when nurses started or completed a task, they pushed the event button and recorded their tasks of the moment by speaking short sentences. These sentences include confirmation messages for nursing

[3] "ENTO" means "to discharge from hospital," originating from the German word "Entlassen.".

tasks, conversations with patients, and so on. First, we transcribed these short sentences and related them to nursing tasks associated with each one. To effectively build the Nursing Task Corpora, we also annotated job categories for each nursing task. We used terms for job categories included in Kango Gyoumu Shishin (看護業務指針) published by the Japanese Nursing Association (日本看護協会) [11] and Classification of Nursing Practice (看護行為用語分類) published by the Japan Academy of Nursing Science (日本看護科学学会) [12]. These annotations were tagged by one ex-nurse who had worked in hospitals for three years or longer, just like for making transcriptions. An example from the corpora is shown in Table 2.

Table 2. Example from the Nursing Task Corpora

Time	Utterance	Job Category
11:01:00	I'm going to join a short conference.	18-106 conference
11:20:48	The short conference is finished.	18-106 conference
11:28:11	I'm going to prepare a drip infusion set for Abe-san.	13-63-6A0502 intravenous infusion
11:32:01	I've finished preparing the drip for Abe-san.	13-63-6A0502 intravenous infusion

In this paper, since we focus on task analysis, we will discuss task analysis using the Nursing Task Corpus in the following sections.

5 Results and Discussion of Task Analysis

In this section, we show a task analysis using the spoken corpora, consider the results, and discuss our study in relation to automatic task analysis.

5.1 Term Definition for Task Analysis

To make a task corpus such as that shown in Table 2, we used task names (task terms), explained previously, as job categories. Table 3 shows an example of the job category list. It is difficult to find direct relations among these categories and nursing task corpora in the real field because they have lexical ambiguity, various expressions, and are too detailed for analysis of the relation between nursing tasks and medical malpractice. We defined some task terms for the nursing assignments, and correlated the task terms with the job category list.

The task terms are defined as follows: One-day nursing jobs, called **Daily Nursing Assignments**, are divided into many tasks that are determined by the job categories list etc. Daily Nursing Assignments consist of many **Nursing Tasks**, for example, assistance with meals, instillation of drips, preparation of infusions, and so on. Nursing Tasks are divided further into two categories, i.e.,

Table 3. A Part of the Job Category List

No	Job List 1 (Major Division)	No	Job List 2	Cord No	Term of CNPBook (Minor Division)
1	Hygiene (身体の清潔)	1	Bed-bath (清拭)	2D0301	Hand-bath (手浴)
				2D0302	Foot-bath (足浴)
				2D0601	Body-bath (全身清拭)
		2	Hair-care (洗髪, 整髪)	2D0401	Washinghair (洗髪)
				2D0301	Stylinghair (整髪, 結髪)
	.....				
6	Patient mover (患者の移送)	26	Patient mover	2G0201	Move to wheelchair (車椅子への移乗)
				2G0301	Move with assistive device (補助器具を用いた移動の介助)
				2G0401	Other transfer (移送)
	.....				
13	Doctor's assistant (診療, 治療の介助)	58	Rounds (回診)		
		59	Change bandage (包交)	6C0101	Injurycare (創傷ケア)
		60	Brace (ギプス)		
		63	IVH-care (IVH, 持続点滴の管理)	6A0502	Intravenous infusion (点滴静脈内注射)
	.....				
18	Takeover among nurses (看護師間の報告, 申し継ぎ)	105	Changeover (申し送り)		
		106	Conference (カンファレンス)		
	.....				
34	Control of staff' health care (職員の健康管理)	162	Meal (食事)		
		163	Rest, Break Time (休息, 休憩)		

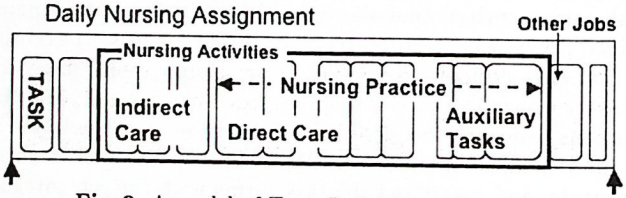


Fig. 2. A model of Term Definition of Nursing Tasks

Nursing Activities and **Other Tasks**. Nursing Activities include jobs that nurses have to do, and are again divided into two categories, i.e., **Nursing Practices** and **Indirect Care**. Nursing Practices include jobs directly related to patients, consist of **Direct Care** and **Auxiliary Tasks** that support doctors. A standardized model of the term definition of nursing tasks is shown in Fig. 2.

The relations of our term definition and the job categories list are shown in Table 4.

Table 4. Relation of the Term Definition and the Job Categories List

Term Definition	No of Job Category List	Job example
Direct Care	1 – 12	Bedside-bath, Care of hospitalization and release
Auxiliary Tasks	13 – 16	Doctor assistance, Vital care, Medical tests
Indirect Care	17 – 27	Report of medical condition, Conference among nurses
Other Tasks	28 – 35	Messenger tasks, Communication with other sections

5.2 Results of Task Analysis from Nursing Task Corpora

We created the Nursing Task Corpora to analyze nursing tasks for the whole day. We annotated job categories for each transcribed sentence, analyzed the working time taken for the four categories (Direct Care, Auxiliary Tasks, Indirect Care, and Other Tasks) using the Nursing Task corpora. Example results are shown in Table 5. These results are average values of task analysis using all data in each shift in one department.

Indirect Care in the Nursing Activities, such as takeover among nurses, and preparing medicines, takes a long time in all shifts. The time for Direct Care and Auxiliary Tasks is unexpectedly short in all shifts. Therefore, we think we can provide some practical methods to concentrate attention on only the time for Direct Care and Auxiliary Tasks.

Table 5. Example Results of Job Analysis using the Nursing Task Corpora

Job Categories	Dayshift	Twilightshift	Nightshift
Direct Care	0:59:06	0:54:12	1:36:27
Auxiliary Tasks	1:40:19	1:26:28	1:53:22
Indirect Care	2:52:11	2:10:53	3:15:50
Other Tasks	0:53:24	0:16:53	1:05:45
Unspecified	1:30:08	0:50:14	1:23:40
Total Time	7:55:08	5:38:39	9:15:04

5.3 Discussion of Task Analysis

The time periods for Direct Care and Auxiliary Tasks are very important ones for preventing medical malpractice because nurses give medical treatment to patients directly. If nurses can focus their attention on these time zones, we think there will be great success in preventing medical malpractice.

To realize this, we need to know which tasks nurses are doing as they do them. This can be done by making transcriptions with the aid of speech recognition tools [13–15]. For the next step, we need methods to analyze nursing tasks automatically. One solution could be applying text classification technologies to task analysis; that is, if transcribed sentences are classified into the same job category, the sentences will include at least one of the same terms. We investigated the features of the task corpora in detail to explore the possibility of applying text classification to the nursing task analysis. We checked about 6,000 utterances in a part of the task corpora as a trial. Thirty-nine nurses participated in the trail experiment. They were divided into 35 job categories within the major divisions[4], such as 1-1-2D0401 (Hygiene/Hair-care/Washing-hair 洗髪), 13-58 (Doctor assistance/Rounds 回診), as shown in Table 3. All transcriptions in the trial data were analyzed by the morphological analysis tool Chasen [16]. The frequency of nouns in the transcriptions was 18,737 times (1,590 words), and the frequency of verbs was 6,321 times (530 words). In fact, one utterance includes one verb and two or three nouns, suggesting that each sentence is short because of spoken language.

In the data, the most frequently occurring verb is し (do). With 1,400 appearances this word appears in every transcription of every job category. Verbs, however, are of limited use for understanding nursing tasks. On the other hand, the most frequently occurring noun is さん (Mr, Miss, Mrs.), occurring 1,948 times. This is also useless for understanding the tasks, but nouns of high frequency are terms that are easy to associate with the Job Categories, such as チェック (Check), カルテ (Medical chart), 引き継ぎ (Takeover). We checked the relation between high-frequency nouns and job categories. The features are shown in Table 6.

We randomly checked the features of each shift in one department, the frequent appearance of nouns divided in the same categories. This means that the surface co-occurrence relations between nouns of utterances and job categories are useful for task analysis. Though our corpora include very few data for one department in a certain hospital for analyzing general nursing tasks, we think it is possible to understand nursing tasks automatically using the surface co-occurrence relations between nouns and job categories.

6 Conclusion

We introduced our project and explained the process of task analysis using the nursing task corpora and showed how to analyze daily nursing assignments efficiently using a wearable computer. We have now obtained 800 hours of voice data

[4] There are about 460 job categories in the minor division.

Table 6. Features of Relations with High-Frequency Nouns and Job Categories

Noun	Frequency	Job Categories	Appearance Rate %
チェック (Check)	410	18-109 情報の整理 (Summary of medical chart)	58.2 (239/410)
カルテ (Medical chart)	353	18-109 情報の整理 (Summary of medical chart)	68.6 (242/353)
引き継ぎ (Takeover)	212	18-105 申し送り (Changeover)	89.6 (190/212)
点滴 (Drip)	254	13-63-6A0502 点滴静脈内注射 (Intravenous infusion)	68.5 (174/254)
検温 (Temperature check)	171	15-74 バイタルチェック (Vital check)	84.8 (145/171)

to produce a 20-hour task corpus and a 50-hour dialogue corpus. As a first step, we have been manually transcribing the spoken corpora, and classifying the job categories. Furthermore, we found a correlation between high-frequency terms and job categories. If we can build or extend the contents of a concept-based, a multiple-expression dictionary, and an abbreviated-expression dictionary, we can produce additional corpora and analyze nursing tasks automatically using technologies of speech recognition, morphological analysis, and classification.

Utilizing the spoken corpora, we can expect to understand nursing assignments in detail and the structure of communication among nurses. As a result, it should be possible to prevent medical malpractices due to miscommunication, and to call nurses' attention to certain tasks correlated with medical malpractice. We think a method that makes spoken corpora can produce nursing reports more easily and accurately than "Time Study Survey" can. In future work we will clarify the method in detail and apply it to many hospitals.

In this paper, we only analyzed data from one department in a hospital. However, we have plans to obtain data from some other departments using our devices. Furthermore, by incorporating detailed spoken corpora with other multiple data, we will pursue the exact cause of medical malpractice in hospitals and undertake work to build a nursing ontology.

Acknowledgments

We would like to thank the nurses for their cooperation in our experiment. This research is funded by the National Institute of Information and Communications Technology.

References

1. L. T. Kohn, J. M. Corrigan, and M.S. Donaldson: "To Err Is Human: Building a Safer Health System." The National Academis Press. (1999).

2. Healthcare Safety Precaution Network Project:
"http://www.mhlw.go.jp/topics/2001/0110/to1030-1.html#2-1"
in Japanese, (2001).
3. Medical Information Center Development Center eds.:
"Japan Nursing Practice Standard Master"
http://www.medis.or.jp/4_hyojyun/kango/kango.html in Japanese.
4. Japanese Nursing Association Nursing Practice International Classification Research Project translated,
"International Classification of Nursing Practice (Japanese version),"
http://icnp.umin.jp/ in Japanese.
5. M. J. F. Gales and S. J. Young: "Robust Continuous Speech Recognition Using Parallel Model Combination," IEEE Transactions on Speech and Audio Processing, Vol. 4, No. 5, pp. 352-359, May 1996.
6. S. Nakamura, K. Takeda, T. Yamada, S. Kuroiwa, T. Nishiura, M. Misumachi, M. Fujimoto, and T. Endo: "AURORA-2J: An Evaluation Framework for Japanese Noisy Speech Recognition"
IEICE Transactions on Information and Systems, Vol. E88-D, No. 3, pp. 535-544, 2005.
7. N. Kuwahara, K. Kogure, N. Hagita, H. Iseki: "Ubiquitous and Wearable Sensing for Monitoring Nurses' Activities." The 8th World Multiconference on Systemics, Cybernetics and Informatics SCI 2004.
8. "IPADIC version 2.7.0 (IPA 品詞体系辞書)"
http://chasen.aist-nara.ac.jp/stable/ipadic/ipadic-2.7.0.tar.gz
9. H. Ozaku, A. Abe, N. Kuwahara, F. Naya, K. Kogure and K. Sagara: "Building Dialogue Corpora for Nursing Activity Analysis" In the proceedings of Sixth International Workshop on Linguistically Interpreted Corpora, (2005).
10. K. Katsuyama, Y. Kouyama, Y. Hirano, K. Mase, and K. Yamauchi: "Doctor-Patient Communication Support Method by Visualizing Topic Structure," In the Japan Journal of Medical Informatics, pp. 578–587, Vol. 24, No. 6, 2004. (in Japanese.)
11. "Kango Gyoumu Kubun Hyou (看護業務区分表),", Kango Gyoumu Kijunsyu (看護業務基準集), pp. 336–337, published by Japan Nursing Association (日本看護協会), ISBN 4-8180-1089-8, 2004.
12. "Classification of Nursing Practice (看護行為用語分類)," Japan Academy of Nursing Science (日本看護科学学会), ISBN 4-8180-1142-8, 2005.
13. S. Nakamura: "Towards Robust Speech Recognition to Acoustic Disturbances." In the Journal of the Acoustical Society of Japan, pp. 662–667, Vol. 57, No. 10, 2001. (in Japanese.)
14. Julius – open source real-time large vocabulary speech recognition engine: "http://julius.sourceforge.jp/en/index.html"
15. M. Fujimoto and S. Nakamura: "Particle Filtering and Polyak Averaging-based Non-stationary Noise Tracking for ASR in Noise," In the Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU05), pp. 337–342, Nov. 2005.
16. Chasen 2.3.3 "http://chasen.naist.jp/hiki.Chasen/"

Information Retrieval

Improved Focused Web Crawling Based on Incremental Q-Learning

Yunming Ye[1], Yan Li[1], Joshua Huang[2], and Xiaofei Xu[1]

[1] Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China
yym_sjtu@yahoo.com.cn
[2] E-Business Technology Institute, The University of Hong Kong, Hong Kong
jhuang@eti.hku.hk

Abstract. This paper presents *IQ-Learning*, a new focused crawling algorithm based on incremental Q-learning. The intuition is to extend previous reinforcement learning based focused crawler with the incremental learning mechanisms so that the system can start with a few initial samples and learns incrementally from the knowledge discovered online. First, a sample detector is used to distill new samples from the crawled Web pages, upon which the page relevance estimator can learn an updated estimation model. Secondly, the updated page relevance information is fed back to the Q value estimator constantly when new pages are crawled so that the Q value estimation model will be improved over time. In this way, the reinforcement learning in focused crawling becomes an incremental process and can be more self-adaptive to tackle the complex Web crawling environments. Comparison experiments have been carried out between the *IQ-Learning* algorithm and other two state-of-the-art focused crawling algorithms. The experimental results show that *IQ-Learning* achieves better performance in most of the target topics.

Keywords: Focused Crawler; Q-Learning; Incremental Learning; Web

1 Introduction

A Web crawler is an information gathering system that traverses the Web by following the hyperlinks from page to page and downloads Web pages that are interested. General Web crawlers visit the Web in an unselective mode. They aim at collecting Web pages as many as possible to build search engines. Different from the general crawler, a focused crawler [1] is an intelligent Web crawler that traverses the Web selectively to download pages in some predefined target topics. Given a search topic and a predefined maximum download number, the goal of a focused crawler is to collect as many relevant pages as possible while retrieving as fewer irrelevant pages as possible in the crawling process.

Focused crawlers are very useful in several Web applications [2], such as collecting Web pages with specific topics for domain-specific search engines, archiving specific page collections for a digital library, and gathering systematic information in some specific topics for market research or survey of literature on the

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 139-150

Web for a scientific research. Therefore, it has attracted much attention in recent years. Several useful methods for building focused crawlers have been proposed, such as the PageRank value based crawling strategy [3], reinforcement learning [4], the evolutionary multi-agent system [5], the statistical modeling method [6], the 'critic-apprentice' learning paradigm [7], to name a few.

Among previous works, reinforcement learning has been testified to be a very effective method for focused crawling [4, 7]. From an agent perspective, a focused crawler is an intelligent agent that interacts with the environment (the Web) and attains goal states (picks up relevant pages). It's natural to fit the crawling process into a reinforcement learning framework: the crawling process is to search an optimal decision making strategy which can be modeled as a discounted function of state, action and reward, where the number of relevant pages collected is the state of the crawler, following a hyperlink is an action, and finding a relevant page is an immediate reward. In [4], Rennie and et al. gave a first solution to design a focused crawler based on reinforcement learning, and their experiments showed good results. However, their method has two limitations:

1) the reward function is learned offline from manually labeled sample pages, which is too time-consuming as the training set may contain thousands of pages that collecting and labeling of them is costly;

2) as the action-reward model is built by offline learning and the model stays unchanged during crawling, the static learned models cannot quickly adapt to the diverse Web environment where the content of Web pages as well as the link structure for different topics are very diverse, and different target topics may require different crawling strategies. The essence of these problems lies in the lack of incremental learning capabilities.

In this paper, we present a new focused crawling algorithm *IQ-Learning*, to enable the focused crawler to learn incrementally from online crawling information so that the reward-action model can be adapted and improved over time. In its incremental *Q-learning* learning framework, the crawler system learns incrementally from the knowledge discovered online through two mechanisms. First, a sample detector is used to distill new page samples from the crawled pages, upon which the page relevance estimator can learn an updated estimation model. Secondly, the updated page relevance information is fed back to the Q value estimator constantly when new pages are crawled so that the Q value estimation model will be improved over time. With incremental learning, the algorithm can start from a few initial samples and be self-adaptive to different crawling environments. Moreover, the reinforcement learning framework also endows it with the ability to make tradeoff between exploiting immediate rewards and exploring future rewards efficiently.

We have compared *IQ-Learning* algorithm to other state-of-the-art focused crawling algorithms. Experimental results showed that *IQ-Learning* got better performance in English topics as well as Chinese topics. It's also testified that the combination of incremental learning capability with reinforcement learning is a very effective approach to improving the performance of focused crawlers.

The rest of this paper is organized as follows. In section 2, we formalize focused crawling in a Q-learning framework. Section 3 describes the details of *IQ-Learning*. Experimental results and analysis are presented in Section 4. In Section 5 we draw some conclusions about our work and point out some directions for future work.

2 Focused Crawling as Q-learning

In [4], Rennie and et al. have presented a framework to formalize focused crawling as reinforcement learning. This section gives a brief summary of the main idea, and discusses the disadvantages of Rennies's method that can't perform incremental learning during the crawling process.

Reinforcement learning addresses the problem of how an autonomous agent can learn to choose optimal actions to achieve its goals through trial-and-error interactions with a dynamic environment [8]. Unlike supervised learning, its learner is never told the correct action for a specific state, but is simply told how good and bad the selected action is, expressed in the form of a scalar 'reward'. In the standard reinforcement learning model, the task of the agent can be defined as follows: the agent exists in an environment with a set of possible states S, and it can perform a set of actions A; when performing an action the state of the agent will change to another state defined by a transition function as: $\delta : S \times A \longrightarrow S$, and for each action the agent will get a reward defined by a reward function as: $r : S \times A \longrightarrow R$, the task of the agent is to learn a policy $\pi : S \longrightarrow A$ that maximizes the sum of its rewards over time (when the agent takes a set of actions).

Q-learning [8] is one of the most popular learning algorithms in reinforcement learning. It simplifies the policy learning as the estimation of the Q value for each state-action pair.The value $Q(s, a)$ (namely Q value) is defined to be the expected discounted sum of future rewards obtained by taking action a from state s and following an optimal policy thereafter. Once these values have been learned, the optimal action from any state is the one with the highest Q value. [8] gives a detailed explanation of Q-learning.

The goal of a focused crawler is to selectively seek out pages that are relevant to the predefined target topics. An ideal focused crawler retrieves the maximal set of relevant pages while simultaneously traversing the minimal number of irrelevant documents on the Web. From an agent perspective, a focused crawler is an intelligent agent that interacts with the Web environment to achieve its goal of getting maximum relevant pages. It's natural to fit the crawling process into a Q-learning framework: the crawling process is to search an optimal decision making strategy which can be modeled as a discounted function of state, action and reward, where the number of relevant pages collected is the state of the crawler, following a hyperlink is an action, and relevant page are immediate rewards (Q values).

To fit focused cawling in the Q-learning framework, two approximations are needed [4]: (1) the number of relevant pages on the Web is infinite and the state

is independent of it; (2) the distinction between the hyperlinks can be captured by the texts in the neighborhood of the hyperlinks, that is, the action is described as the link context of the corresponding hyperlink. With the first approximation, the state space becomes one single state, otherwise it will contain 2^n states, a very huge and insoluble number, where n is the actual number of relevant pages.

Based on the two approximations, the task of focused crawling in Q-learning framework is to follow a list of actions (hyperlinks) such that the cumulative Q value (the number of relevant pages) can be maximized. This can be further divided into three steps based on the two approximation as: (1) assigning the appropriate Q value to each hyperlink in the training set; (2) build Q value estimation model from the sample pair of hyperlink and its corresponding link context (such as the anchor text); (3) using the estimation model to compute a Q value for each candidate hyperlink (URL) in the URL queue to be crawled, where the Q value represents the crawling priority for the candidate URL.

In previous work, the step of assigning Q value is done by manually labeling which is too time-consuming and inefficient. Moreover, the estimation model remains the same and can't be self-adaptive in the crawling process. This is problematic for two reasons: (1) the initial samples for training the model are never sufficient to build an accurate model; (2) as the content of Web pages as well as the link structures for different topics are very diverse, and different target topics may require different crawling strategies, the static learned models cannot quickly adapt to the diverse Web environment. These problems are addressed in the *IQ-Learning* crawling algorithm, which effectively combines the incremental learning with *Q-learning*.

3 Focused Crawler Based on Incremental Q-Learning

3.1 An Incremental Q-Learning Framework

The *IQ-Learning* crawling algorithm is based on an incremental *Q-Learning* framework, as shown in Fig.1. In this framework, the crawling module gets URLs from the URL queue and downloads (crawls) the corresponding Web pages through HTTP/HTTPS protocols. The crawled pages will be analyzed to extract URLs which will be added to the URL queue as new candidate URLs to be crawled. Similar to general focused crawlers, the framework uses a page relevance estimator (page classifier) and a link predictor to make the crawler ***focus*** on the target topic (i.e., crawling selectively to get maximum relevant pages). The relevance of the crawled pages are computed by the page relevance estimator, which employs a TF*IDF [9] relevance evaluation model to compute a continuous relevance value for each crawled page. The Q value learner has two functions: one is to compute the Q value for each pages in the training set based on the relevance information provided by the page relevance estimator, the other is to build the Q value estimation model from the training samples through the mapping between link contexts and Q values. According to the Q value estimation model, the link predictor computes and assigns a Q value to each URL in the URL queue, where the Q values denote the visiting priority

(i.e. the probability of being relevant) for the candidate URLs, and the URLs with higher Q values will be crawled first.

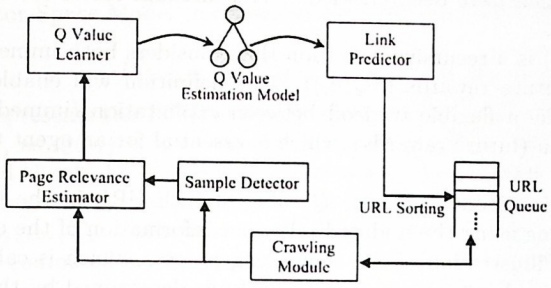


Fig. 1. An incremental *Q-Learning* framework for focused crawling.

There are two incremental learning processes in this framework. First, the page relevance estimator is an incremental learner. It is based on the 'soft' classification method from our work [10]. It needs only a few initial training samples and uses a sample detector to analyze the crawled pages to distill new samples automatically. The new samples are fed back to the page relevance estimator to learn a new relevance estimation model incrementally. This process will improve the accuracy of page relevance estimation over time, which has been testified in the previous work [10]. Details about this process were discussed in [10, 11].

Secondly, the Q value learner (and in turn the link predictor) also works in an incremental manner. Initially, the learner has a very limited training set. As the crawling proceeds, the crawled pages and their relevance information will be transferred to the Q value learner as the new learning resources. New training samples will be distilled based on predefined relevance thresholds. For 'old' training samples, their Q values will be updated as well. The Q value learner will learn from the training set, which is updated constantly during crawling, to improve its estimation model incrementally.

3.2 The Definition of An Updatable Q Value

The Q value for each new URL in the URL queue represents the crawling priority of the URL. Given a sample hyperlink (URL), the Q value learner computes the corresponding Q value according to the function defined as follows:

$$Q(u_i) = R(d_i) + \gamma Q(d_i), \qquad Q(d_i) = \frac{1}{n} \cdot \sum_{k=1, u_k \in d_i}^{n} Q(u_k) \tag{1}$$

where u_i is a crawled URL, d_i is the corresponding page for u_i. $Q(u_i)$ is the Q value for u_i, which is a dynamic value updated constantly during crawling.

$R(d_i)$ is the relevance of d_i with regard to the target topic, and is calculated by the page relevance estimator. u_k is a crawled child URL of d_i, that is, d_i links to the page corresponding to u_k. The value n is the total number of the children URLs of d_i which have been crawled. γ is a discount factor for converting the future rewards.

Formula (1) is a recursive definition that considers both immediate rewards ($R(d_i)$) and future rewards ($\gamma Q(d_i)$). This definition will enable the focused crawler to make a flexible tradeoff between exploitation (immediate rewards) and exploration (future rewards), which is essential for an agent to achieve the global optimal state.

According to formula (1), the Q value of each URL in the training set is calculated online using the updated relevance information of the crawled pages. Fig.2 gives an illustration of the computing process, which is called 'relevance feedback'. For each sample page, its Q value is determined by the Q values of its children pages recursively. When new pages are crawled (or the relevance values of the pages have been updated), the new relevance information will be propagated back to their ancestor pages along the link paths so that the ancestor pages can re-compute their Q values agilely. For instance, in Fig.2 the link path for page G (corresponding to the URL u) is $A \longrightarrow B \longrightarrow E \longrightarrow G$. The left part of Fig.2 denotes the link graph before the page G is crawled, while the right part shows the link graph after visiting G. During the feedback, the relevance of G is back-propagated along the link path $G \longrightarrow E \longrightarrow B \longrightarrow A$, so that the Q value for each page on the path is recalculated according to formula (1).

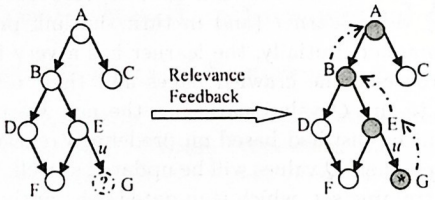


Fig. 2. An illustration of the relevance feedback along link path.

3.3 Mapping Link Context to Q Value

After computing the Q values for the sample URLs in the training set, the Q value estimator needs to build the Q value estimation model that can be used to predict the Q value for a candidate URL given the link context of the URL. The prediction of Q value is a regression problem because the Q value is continuous. Directly solving this regression problem is difficult as training samples are limitted. Similar to Rennies's method [4], we solve this problem by casting the regression problem into a classification prolbem through the discretization of

the Q value [8], and employ the Naive Bayes (NB) as the classification method to map link contexts into the discretized Q values.

For each training sample $\langle \Gamma(u_i), Q(u_i) \rangle$, the link context $\Gamma(u_i)$ for URL u_i is constructed by concatenating all the anchor texts from its parent pages. We use the Vector Space Model to represent $\Gamma(u_i)$ as a TF*IDF vector as:

$$\Gamma(u_i) = \langle \omega_{1i}, \omega_{2i}, \ldots, \omega_{ki}, \ldots, \omega_{ni} \rangle \tag{2}$$

where ω_{ki} represents the TF*IDF weight of the term t_k in the link context of u_i.

However, as $Q(u_i)$ is a continuous value, it can't be used directly by NB classifier and should be discretized and normalized as follows: all Q values are normalized into an interval [0,1] which is then discretized into k sub-intervals as: $C_1, C_2, \ldots C_i, \ldots, C_k$. Each sub-interval C_i represents a class, and the Q value of a class is computed as the average value of all the Q values in the sub-zone. After the discretization, the initial training sample $\langle \Gamma(u_i), Q(u_i) \rangle$ is represented as $\langle \Gamma(u_i), C_i \rangle$.

NB classifier is a probabilistic classifier based on the statistical independence assumption that features in learning instances are considered conditionally independent given the target label. For our mapping problem, the task of the classifier is to predict the class C^* for the link context of a candidate URL $\Gamma(u_i)$, where the Q value of the class C^* will be assigned to the URL u_i. In NB classifier, this can be formalized as: to find a class C^* for $\Gamma(u_i)$ such that the conditional probability $P(C^*|\Gamma(u_i))$ will be maximized:

$$C^* = \arg\max_{c_j} P(C_j|\Gamma(u_i)) = \arg\max_{c_j} P(C_j)P(\Gamma(u_i)|C_j) \tag{3}$$

Directly computing $P(\Gamma(u_i)|C_j)$ is difficult since the number of possible vectors for $\Gamma(u_i)$ is too high. However, according to the independence assumption of NB that the occurrence of term t_k in a link context is independent of the occurrence of any other term, the formula (3) can be replaced by formula (4) as:

$$C^* = \arg\max_{c_j} P(C_j)P(\Gamma(u_i)|C_j) = \arg\max_{c_j} P(C_j) \prod_{k=1}^{|\Gamma(u_i)|} P(\omega_{kj}|C_j) \tag{4}$$

Thereby, learning a NB classifier is to estimate $P(C_j)$ and $P(\omega_{kj}|C_j)$ from the training set. $P(C_j)$ can be calculated as the fraction of the Q value number of class C_j in the entire training set. The posterior probability $P(\omega_{kj}|C_j)$ is calculated as:

$$P(\omega_{kj}|C_j) = \frac{1 + \sum_{i=1}^{|c_j|} \omega_{ki}}{|V| + \sum_{i=1}^{|c_j|} \omega_i} \tag{5}$$

where $|V|$ is the vocabulary size and serves for Laplace smoothing of the non-occurring terms, ω_{ki} is the total TF*IDF weight of term t_k that occurs in the link context $\Gamma(u_i)$ of class C_j, and ω_i is the total weight of all term occurrences in the link context $\Gamma(u_i)$ of class C_j.

According to the above definitions, the NB learner will be trained incrementally during crawling, and the link predictor in Fig.1 uses the learned NB classification model to compute the Q value for each candidate URL as: the classfier classifies a candidate URL u_i into a certain class C^*, where the corresponding Q value of the class C^* is assigned to u_i as its Q value.

3.4 The Algorithm

The pseudo-code of *IQ-Learning* crawling algorithm is presented in Fig.3. The focused crawler starts from some seed URLs to crawl the Web selectively. At the initial stage, the page relevance estimator and Q value estimator need only a few training samples, which the user can easily provide. As the crawling process proceeds, new samples will be distilled automatically and fed back to the learners to build new estimation models, as shown in the step 11 and 13. As the page content and the link structure on the Web are diverse, this self-adaptive functionality will enable the focused crawler to learn an optimal crawling strategy quickly, so that it will not be restricted to the limited initial samples, and will greatly improve the performance and robustness of the system.

Ideally, the incremental learning should be performed continuously. However, as the learning process is computationally expensive, in practical implementations the crawler always executes the learning process in a batch mode, that is, every X pages (where X can be 100). This is a tradeoff between the crawling accuracy and the efficiency.

Input: K-keywords; W^*-initial samples; S-seed URLs;
maxDownload-maximum pages to be crawled
Output: D- the crawled page repository
1. Insert S into the URL queue H as the initial URLs for the crawling module;
2. Page relevance estimator learns from w^* initially;
3. while ((D.size$\leq$ ***maxDownload***) and (H.size$>$ 0)) do:
4. Fetch the URL u with highest Q value from H;
5. Crawl the page d corresponding to u;
6. $D = D \bigcup d$;
7. Extract new URLs U^* from d;
8. Insert U^* into H;
9. Page relevance estimator computes the relevance of d: $R(d)$;
10. Feed back $R(d)$ along the link paths to recalculate the Q values for the ancestor pages of d based on formula (1);
11. Re-training the naïve Bayes Q-value-mapping classifier with the new Q values got from step 10;
12. Using the new classifier to re-estimate Q values for candidate URLs in H;
13. Page relevance estimator learn incrementally from d using sample detector;
14. Sort the URL queue H based on the new Q values;
15. end while

Fig.3. The pseudo-code for the *IQ-Learning* algorithm.

4 Experiments

4.1 Experiment Setup

We have implemented the *IQ-Learning* algorithm in our focused crawler system *iSurfer*. A high-performance crawling module was used to download Web pages from the Web. The system also included an efficient HTML document parser to parse and analysis the pages. The well-known Porter stemming algorithm was used to stem English words. To experiment on Chinese Web pages, we used Maximized-Matched Chinese word segmentation algorithm to extract Chinese words from Chinese text blocks, as unlike English text there is no explicit delimiters like space between Chinese words.

Target topics were defined by the keywords and a few sample pages in our experiments. The initial samples were obtained through meta-searching the Yahoo! search engine. Keywords were sent to Yahoo! search, and the result pages were parsed to extract the hit URLs, which were presented to the system as the seed URLs as well as the initial samples for page relevance estimator.

In traditional information retrieval research, the precision and recall are the main evaluation metrics. However, it is difficult to get the recall for focused crawlers, as measuring the size of relevance set from the enormous Web is almost impossible. We employ the precision, which is named *'harvest rate'* in focused crawler community [1], as the main performance metric. Harvest rate is defined as the percentage of the crawled pages that are relevant w.r.t the target topics.

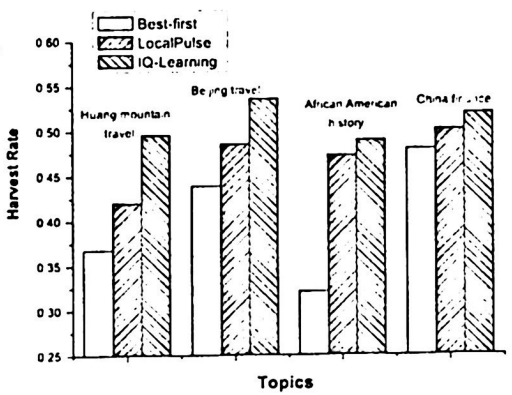
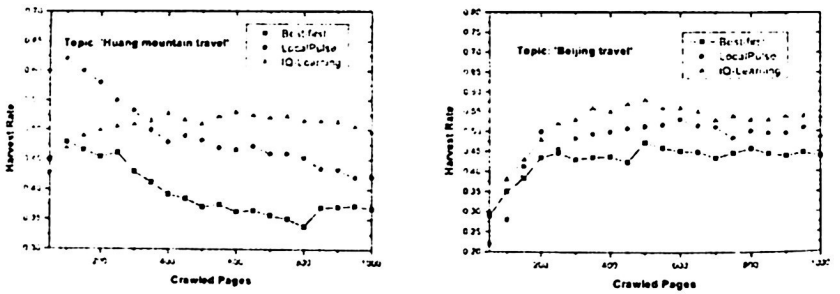


Fig. 4. The final harvest rates for the three algorithms.

4.2 Comparison Experiments

In our experiments, we compared *IQ-Learning* to the other two state-of-the-art focused crawling algorithms: (1) the best-first algorithm [12], which has been testified as one of the best-performed crawling strategies in previous comparison studies; (2) the *LocalPulse* algorithm [10, 11], which was used in *iSurfer* previously. *LocalPulse* has incremental learning capability, but it's not based on reinforcement learning, and it can't perform online relevance feedback along the link path, which affects its self-adaptive ability. The three algorithms were tested on more than 50 topics covering a variety of domains. These topics had diverse topical broadness, both in English and in Chinese. For each topic, crawlers started with 10 seed sample URLs.

Fig.4 shows the final harvest rates of the algorithms on some topics after crawling one thousand pages. Both *IQ-Learning* and *LocalPulse* outperformed the best-first crawler in all topics. This illustrates that incremental learning is very effective for improving the performance of focused crawler. Another important observation was that the *IQ-Learning* got higher harvest rate than *LocalPulse* although both of them can learn incrementally in the crawling process. To explain this phenomenon, we investigated their harvest rates during crawling.



(a) (b)

Fig. 5. The dynamic harvest rate movement of the three algorithms.

Fig.5 shows the dynamic harvest rates of the three algorithms on the Chinese topics 'Huang Mountain Travel' and 'Beijing Travel'. At the early stage of the crawling, *LocalPulse* got more relevant pages than *IQ-Learning*. However, as the crawling proceeds, *IQ-Learning* catch up with *LocalPulse* very quickly. This is due to the fact that *LocalPulse* employs a greedy strategy in selecting candidate URLs, which always focuses on the crawling area with more immediate rewards and does not consider future rewards, therefore it is very easily to stick into local optimal states. On the contrary, *IQ-Learning* makes a good tradeoff

between immediate rewards and future rewards, through which the crawler can get more relevant pages in the long run and can achieve global optimal states. This testifies that combining reinforcement learning with incremental learning is a more effective method for focused crawling.

4.3 The Tradeoff between Immediate Rewards and Future Rewards

According to formula (1), the parameter γ is a bias factor between the immediate rewards and future rewards. The value of it will greatly affect the performance of the system. We used different values of γ to do the crawling experiments. Fig.6 shows the experimental results on the topic 'Huang Mountain Travel'.

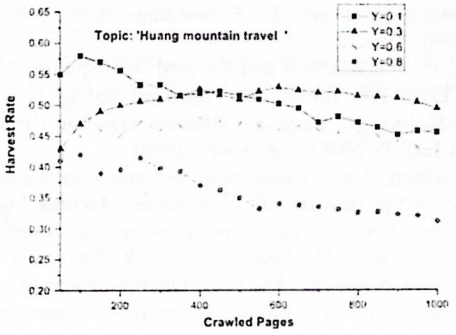


Fig. 6. The effect of the discount factor γ on *IQ-Learning*.

As the figure illustrates, the harvest rate will become higher in the early stage of the crawling process if γ is set to a smaller value, such as $\gamma = 0.1$ in Fig.6, which means the crawler is more 'greedy' and biased for the candidate URLs with immediate rewards. However, as the crawling process continues, the ones with bigger value of γ get higher performance because their early consideration of future rewards is paid for soon. On the other hand, the value of γ can't be set too high as well. Otherwise, the future rewards will be over-weighted w.r.t the immediate rewards, and the focused crawler may fall into topic-drift. As the figure shows, when γ is set to 0.8, the harvest rate of the crawler drops dramatically. In our experiences, a value between 0.3 0.4 for γ is a proper choice.

5 Conclusions

This paper presents *IQ-Learning*, a new focused crawling algorithm based on incremental *Q-learning*. Both the page relevance estimator and the Q value estimator are endowed with incremental learning ability to learn improved models

from the online crawled pages over time. With incremental learning, *IQ-Learning* can start with a few initial samples and self-adapt to diverse crawling environments, which will result in a more optimized crawling strategy. Our experimental results testifies that the combination of incremental learning with reinforcement learning is effective for improving the performance of focused crawler.

In the future, we plan to extend our work with a knowledge base which will enable the system to accumulate crawling knowledge from different crawling processes. We will also employ semantic-based Web page segmentation algorithms to enrich the link context with more relevant neighborhood text.

References

1. Chakrabarti, S., M., V.D.B., Dom, B.: Focused crawling: a new approach to topic-specific web resource discovery. In: Proceedings of the 8th International WWW Conference. (1999)
2. Chau, M., Chen, H.: Personalized and Focused Web Spiders. In Ning Zhong, Jiming Liu, Yiyu Yao (Eds), Web Intelligence. Springer-Verlag, New York. (2003)
3. Cho, J., Garcia-Molina, H., Page, L.: Efficient crawling through url ordering. In: Proc. of the 7th Intl. WWW Conference. (1998)
4. Rennie, J., McCallum, A.K.: Using reinforcement learning to spider the web efficiently. In: Proc. of the 16th Intl. Conference on Machine Learning. (1999)
5. Menczer, F., Belew, R.K.: Adaptive retrieval agents: Internalizing local context and scaling up to the web. Machine Learning **39** (2000) 203–242
6. Aggarwal, C.C., Al-Garawi, F., Yu, P.S.: On the design of a learning crawler for topical resource discovery. ACM Transactions on Information Systems, **19** (2001) 286–309
7. Chakrabarti, S., K., P., M., S.: Accelerated focused crawling through online relevance feedback. In: Proc. of the 11th Intl. WWW Conference. (2002)
8. Mitchell, T.: Machine Learning. McGraw-Hill, New York. (1997)
9. Baeza-Yates, R., Ribeiro-Neto: Modern Information Retrieval. ACM Press Series/Addison Wesley,, New York (1999)
10. Ye, Y., Ma, F., Lu, Y., Chiu, M., Huang, J.: Isurfer: A focused web crawler based on incremental learning from positive samples. In: Proceedings of the 6th Asia-Pacific Web Conference. (2004)
11. Ye, Y.: Topic-driven Web Resource Discovery: Models, Algorithms and Applications. PhD Thesis, Shanghai Jiao Tong University, Shanghai (2004)
12. Menczer, F., Pant, G., Srinivasan, P.: Topical web crawlers: Evaluating adaptive algorithms. ACM Transactions on Internet Technology **4** (2004) 378–419

Automatic Identification of Chinese Stop Words

Feng Zou, Fu Lee Wang, Xiaotie Deng, Song Han

Computer Science Department, City University of Hong Kong
Kowloon Tong, Hong Kong
phenix@cs.cityu.edu.hk, {flwang, csdeng, shan00}@cityu.edu.hk

Abstract. In modern information retrieval systems, effective indexing can be achieved by removal of stop words. Till now many stop word lists have been developed for English language. However, no standard stop word list has been constructed for Chinese language yet. With the fast development of information retrieval in Chinese language, exploring Chinese stop word lists becomes critical. In this paper, to save the time and release the burden of manual stop word selection, we propose an automatic aggregated methodology based on statistical and information models for extraction of the stop word list in Chinese language. The novel algorithm balances various measures and removes the idiosyncrasy of particular statistical measures. Extensive experiments have been conducted on Chinese segmentation for illustration of its effectiveness. Results show that the generated stop word list can improve the accuracy of Chinese segmentation significantly.

1. Introduction

In information retrieval, a document is traditionally indexed by words [10, 11, 17]. Statistical analysis through documents showed that some words have quite low frequency, while some others act just the opposite. For example, words "and", "of", and "the" appear frequently in the documents. The common characteristic of these words is that they carry no significant information to the document. Instead, they are used just because of grammar. We usually refer to this set of words as stop words [10, 11, 21].

Stop words are widely used in many fields. In digital libraries, for instance, elimination of stop words could contribute to reduce the size of the indexing structure considerably and obtain a compression of more than 40% [10]. On the other hand, in information retrieval, removal of stop words could not only help to index effectively, but also help to speed up the calculation and increase the accuracy [20].

Lots of stop word lists have been developed for English language in the past, which are usually based on frequency statistics of a large corpus [21]. The English stop word lists available online [22, 23] are good examples. However, no commonly accepted stop word list has been constructed for Chinese language. Most current researches on Chinese information retrieval make use of manual or simple statistical stop word lists [1, 2, 3], some of which are picked up based on the authors experiences consuming a lot of time. The contents of these stop lists vary a lot from each other. With the fast

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 151-162

growth of online Chinese documents and the rapid increase of research interest in Chinese information retrieval, constructing a general Chinese stop word list together with an applicable generating methodology becomes critical. In order to save the time and release the burden of manual stop word selection, an automatic aggregated methodology would be a better choice.

One of the difficulties for automatical identification of stop words in Chinese language is the absence of word boundaries. Different from texts in English and other western languages, which are segmented into words by using spaces and punctuations as word delimiters, Asian languages, such as Chinese, do not delimit words by space. Usually a Chinese word consists of more than one character and the number of characters contained varies. Meanwhile, Chinese characters carry a lot of different meanings. They could be interpreted differently when used together with different characters. The character "的", which is equivalent to the word "of" in English, is taken as an example. It could carry a different meaning in the combination with different characters, such as "的确"(certainly), "的士"(taxi), etc.

Although identification of stop words is quite a hard work without a correct segmentation of text [4], stop words play an important role during segmentation. In English, as an instance, texts are segmented into phrases with the help of stop words and punctuations. Researches show that using effective stop word list can improve the accuracy of Chinese segmentation as well [4].

In our paper, we propose an automatic aggregated methodology for construction of the stop word list in Chinese. Stop words are extracted from TREC 5 and 6 corpora which are widely accepted as standard corpora for Chinese processing. The stop word list is extracted based on statistical and information models. The statistical model extracts stop words based on the probability and distribution. The information model measures the significance of words by using information theory. Results from these two models are aggregated to generate the Chinese stop word list.

To demonstrate the effectiveness of the stop word list, we propose a novel Chinese segmentation algorithm based on it. Experiment has been conducted using the dataset of a recent competition on Chinese segmentation [15]. Results have shown that the stop word list can improve the accuracy of Chinese segmentation significantly. The outstanding performance illustrates the effectiveness of our Chinese stop word list.

The rest of the paper is organized as following. Section 2 covers the methodology for the discovery of the Chinese stop word list. Section 3 analyzes the result of the stop word list extraction experiments. To better prove the effectiveness of this methodology, in Section 3, we also propose an application of the stop word list in the field of segmentation, which is an important step for Chinese information retrieval. Section 4 paints the conclusion.

2. Construction of the Stop Word List in Chinese

Stop words, by definition, are those words that appear in the texts frequently but do not carry significant information. As a result, we propose an aggregated model to measure both the word frequency characteristic by statistical model and its information characteristic by information model. A proper segmentation of Chinese

texts is required before construction of, because the word boundaries are not clear in Chinese texts. In this section, the texts in a large corpus of Chinese documents are first segmented, and then a standard Chinese stop word list is constructed based on the aggregated model.

2.1 Word segmentation

The difficulty of Chinese word segmentation is mainly due to the fact that no obvious delimiter or marker can be observed between Chinese words except for some punctuation marks. Segmentation methods existing for solving this problem of Chinese words include dictionary-based methods [18], statistical-based methods [8]. Other techniques that involve more linguistic information, such as syntactic and semantic knowledge [7] have been reported in the natural language processing literature. Although numerous approaches for word segmentation have been proposed over the years, none has been adopted as the standard. Since segmentation is not the main objective in our methodology, in our paper, we focus on a statistical approach using mutual information, called the boundary detection segmentation, which has been already proved to be effective [19].

Mutual information is to calculate the association of two events. In Chinese segmentation, mutual information of two characters shows how closely these characters associated with each another. Equation (1) shows the computation of mutual information of bi-grams "AB", where P(A,B) denotes the joint probability of two characters, and P(A), P(B) denote probabilities of character 'A' and 'B' respectively.

$$I(A,B) = \log_2\left(\frac{P(A,B)}{P(A)\times P(B)}\right) \tag{1}$$

If the characters are mutually independent, P(A, B) equals to P(A)×P(B), so that I(A, B) equals 0. If 'A' and 'B' are highly correlated, I(A,B) will have a high value.

We first calculate the bi-grams and tri-grams mutual information of all the characters in documents. Based on these values and the change of values of the mutual information in one sentence, one can detect the segmentation points with the threshold value chosen manually.

2.2 Statistical model (SM)

A statistical analysis was conducted on a corpus of 423 English articles in TIME magazine (total 245,412 occurrences of words), top 50 words of which with their frequency are shown in Table 1. Stop words are ranked at the top with much larger frequency than the other words. On the other hand, stop words are also those words with quite a stable distribution in different documents. Statistics of the distribution of word frequencies in different documents (Table 2) offers a good demonstration. A combination of these two observations redefines the stop words as those words with stable and high frequency in documents.

Traditional models extract stop words only based on the accumulated frequency without considering the distribution of words among documents. With the statistical results illustrated in table 1 and table 2, we propose to extract the stop words according to the overall distribution of words. Since mean and variance are two important measurements of a distribution, we extract stop words based on these two criteria.

Table 1. Top 50 words with their frequencies from 423 short TIME magazine articles (245,412 word occurrences, 1.6 MB)

Word	Freq.	Word	Freq.	Word	Freq.	Word	Freq.	Word	Freq.
The	15861	his	1815	U	955	Were	848	Britain	589
Of	7239	Is	1810	Had	940	Their	815	when	579
To	6331	he	1700	Last	930	Are	812	out	577
A	5878	As	1581	Be	915	One	811	would	577
And	5614	on	1551	Have	914	Week	793	new	572
In	5294	by	1467	Who	894	They	697	up	559
That	2507	At	1333	Not	882	Govern	687	been	554
For	2228	It	1290	Has	880	All	672	more	540
Was	2149	from	1228	An	873	Year	672	which	539
With	1839	but	1138	S	865	Its	620	into	518

Table 2. Top 50 words with their variances from TREC 5 English corpus

Word	Var.	Word	Var.	Word	Var.	Word	Var.	Word	Var.
The	33.04	Type	82.14	At	111.7	Their	150.2	Hi	186.6
To	47.94	Language	82.58	Have	112.5	Government	155.5	Would	190.4
Of	49.27	With	84.39	Which	116.9	Been	156.2	About	190.7
In	52.32	By	85.24	From	119.8	But	156.5	More	195.4
A	57.15	article	88.06	Not	120.1	Other	157.5	After	196.0
And	58.55	It	90.13	Will	120.5	All	162.9	Between	196.5
On	70.84	Be	93.87	Was	125.4	Country	163.4	Up	197.6
For	75.37	As	95.92	Also	127.6	Who	175.3	There	198.2
That	76.67	An	110.5	English	130.4	Out	184.3	Or	198.7
Is	81.57	Said	111.7	He	143.6	Were	185.2	online	202.4

First, we measure the mean of the probability (*MP*) of each word in individual document. Suppose there are M distinct words and N documents all together. We denote each word as w_j (j=1, … , M) and each document as D_i (i=1, … , N). For each word w_j, we calculate its frequency in document D_i denoted as $f_{i,j}$. However, the document has different length. In order to normalize the document length, we calculate the probability $P_{i,j}$ of the word w_j in document D_i which is its frequency in the document D_i divided by the total number of words in document D_i. For each word w_j, the *MP* among different documents is summarized as following:

$$MP(w_j) = \frac{\sum_{1 \le i \le N} P_{i,j}}{N} \tag{2}$$

An experiment has been conduced on a 153MB Chinese corpus consisting both of People's Daily news and Xinhua news from TREC 5 and 6. The top 10 words with

highest mean of probability are shown in Table 3.

Table 3. Top 10 words with highest mean of *MP*.

Word	Equvilent Word in English	Mean	Variance
的	Of	0.5926	71.56
和	And	0.1324	72.78
在	In	0.1169	72.23
了	-ed	0.1075	72.98
中国	China	0.0784	75.88
一	One	0.0726	74.52
为	For	0.0670	74.01
有	Have	0.0661	79.79
三	Three	0.0535	80.46
等	etc.	0.0491	74.86

Since stop words should have high *MP* as well as stable distribution, the variance of probability (*VP*) of each word is calculated secondly. Based on the calculation of probability, the *VP* is defined by the standard formula:

$$VP(w_j) = \frac{\sum_{1 \le i \le N} (P_{i,j} - \overline{P}_{i,j})^2}{N} \tag{3}$$

The top 10 words with highest Variance of Probability are shown in Table 4.

Table 4. Top 10 words with lowest *VP*.

Word	Equvilent Word in English	Mean	Variance
的	Of	0.5926	71.56
在	In	0.1169	72.23
和	And	0.1324	72.78
了	-ed	0.1075	72.98
为	For	0.0670	74.01
一	One	0.0726	74.52
中	in/middle	0.0523	74.79
上	above/on/up	0.0431	74.80
等	etc.	0.0491	74.86
积极	Active	0.0117	75.04

Intuitively, the probability of a word to be a stop word is directly proportional to the mean of probability, but inversely to the variance of probability. A combination of these two criteria comes to the final formula. We call it the statistical value (*SAT*) of word w_j.

$$SAT(w_j) = \frac{MP(w_j)}{VP(w_j)} \tag{4}$$

With all these values, a descending ordered lists will be generated. Those ranked in

the top will have a larger chance to be considered as stop words in this model.

Table 5. Top 10 words with highest *SAT*.

Word	Equvilent Word in English	Mean	Variance
的	Of	0.5926	71.56
在	In	0.1169	72.23
和	And	0.1324	72.78
了	-ed	0.1075	72.98
为	For	0.0670	74.01
一	One	0.0726	74.52
中	in/middle	0.0523	74.79
等	etc.	0.0491	74.86
上	above/on/up	0.0431	74.80
中国	China	0.0784	75.88

In table 3 and 4, words like "三" (three) and "积极" (active), with only high *MP* or lower *VP*, will not show up at the top of table *SAT*. On the contrary, words like "的"(of) "和"(and) "在"(in) ranked at the top of all the tables, have both high *MP* and low *VP*.

2.3 Information model (IM)

From the viewpoint of information theory, stop words are also those words which carry little information. Entropy, one of the fundamental measurements of information [5], offers us another method for better describing stop word selection.

The basic concept of entropy in information theory is a measure to count that how much randomness is in a signal or in a random event. An alternative way to look at this is to talk about how much information is carried by the signal. As an example, consider some English text, encoded as a string of letters, spaces and punctuation (so our signal is a string of characters). Since some characters are not very likely (e.g. 'z') while others are very common (e.g. 'e') the string of characters is not really as random as it might be. On the other hand, since we cannot predict what the next character will be, it does have some 'randomness' and the randomness of each character will be different. Entropy is a measure of this randomness, suggested by Claude E. Shannon in his 1948 paper [14]. This could easily be applied to the Chinese text processing. Consider the distribution of each word over documents as an information channel. The high the entropy of this information channel is, the less random the character would be in all documents. Thus we measure the information value of the word w_i by its entropy.

The probability $P_{i,j}$ is its frequency in the document D_j divided by the total number of words in document D_j. We calculate the entropy value (H) for word w_i as following:

$$H(w_j) = \sum_{1 \le i \le N} P_{i,j} * \log(\frac{1}{P_{i,j}}) \tag{6}$$

Similarly to statistical model, one ordered list is prepared for further aggregation. The higher entropy the word has, the lower information value of the word is. Therefore, the words with lower entropy are extracted as candidates of stop words.

Table 6. Top 10 words with highest *H*.

Word	Equvilent Word in English	Entropy
的	Of	0.0177
和	And	0.0059
在	In	0.0054
了	-ed	0.0050
中国	China	0.0039
一	One	0.0037
为	For	0.0034
有	Have	0.0034
三	Three	0.0028
中	In/middle	0.0028

In Table 6, words like "三" (three) and "中" (of), have lower *H* compared with those words such as "的"(of) "和"(and) "在" (in), which have high *H*. It is obviously that "的"(of) "和"(and) and "在" (in) would have better chances to be considered as stop word candidates.

2.4 Aggregation

The ordered lists generated according to two models reveal the features of stop words in different manners, which are all quite reasonable. How to get an aggregation of them? What kind of rules could assure the fairness of the final result? One of the popular solutions to it should be Borda's Rule [12], which covers all the binary relations even when many members of a population have a cyclic reference given a set of voters.

The sorted lists from each model $\{S_1, S_2\}$ are treated the voters' preferences and all the words $\{t_1, t_2, \ldots, t_m\}$ are considered as alternatives. Each model gives out a list $\{t_{j,1}, t_{j,2}, \ldots, t_{j,m}\}$ of all words in non-increasing order. We associate the number 1 with the most preferred word $t_{j,1}$ on the list, 2 with the second $t_{j,2}$ and so on. For all the words, we assign to each of them the number equal to the sum of the numbers all the models assigned to it. The ranking of all the words according to these weights is proposed finally.

3. Experiment and Analysis

To demonstrate the effectiveness of our methodology and to achieve a common Chinese stop word list, we experiment with TREC 5 and 6 Chinese corpora, which contain news reports from both Xinhua newspaper and People's daily newspaper. These corpora cover different aspects of our daily life which ensures the general applicability of our stop word list. The comparison of the list generated by our algorithm with an English stop word list shows that the intersection rate is very high. To clearly show the benefit of our stop word list to Chinese information retrieval, an experiment on segmentation using the Chinese stop word list we extracted is proposed. Experiment tells that our Chinese stop word list facilitates the process of word segmentation in Chinese information retrieval by increasing the accuracy of segmentation, which will result in a better performance in retrieval as well.

3.1 Generation of the Stop Word List

Experiments are conducted on a 153MB Chinese corpus consisting both of People's Daily news and Xinhua news from TREC 5 and 6. We eliminate all the non-Chinese symbols in the preprocessing step. Each uninterrupted Chinese character sequence is kept on one line in the transformed data. On the other hand, phrases like "新华社"(Xinhua News Agency), "人民日报"(people's daily) and "完"(end), that are parts of the news' format of Xinhua and People's Daily corpora, are removed. We apply our methodology on these preprocessed documents and collect two ordered lists before aggregation, namely, statistical list and information list. These two lists are aggregated together to generate the final one.

From the viewpoint of linguistics, similar to English stop words, Chinese stop words are usually those words with parts of speech like adjectives, adverbs, prepositions, interjections, and auxiliaries. Adverb "的"(of), preposition "在"(in), conjunction "因为" (because of) and "所以"(so) are all examples. According to different domains, we could classify all stop words into two categories. One kind is called "generic stop words", which are stop words in the general domain.Another kind is document-dependent stop words. We call them "domain stop words". For example, words "Britain" and "govern" in the Zipf list (Table 1) are not included in most generic stop word list, because they are domain stop words of TIME magazine. That's why in our preprocessing, we eliminated those words such as "新华社" (Xinhua News Agency), which are domain stop words in our news articles.

3.2 Comparison of English and Chinese Stop Word Lists

We compare of our Chinese stop word list with a general English stop word list in table 7. We find that most of the Chinese stop words have corresponding words in English stop word list. For example, word "的"(of) with "of", "和"(and) with "and". However, the specialty of sChinese stop word lists is that some words might have the same meaning, like "和"(and) and "与" (and), both of which means "and". Another

aspect worth mention is that Chinese stop word lists should be treated differently compared with English stop word lists. As known, the meaning of Chinese word might change a lot according to the neighbors. This phenomenon changes the usage of Chinese stop word lists a little bit. Recommended usage of Chinese stop word lists here in further task is to use a factor weakening the weight of these words instead of eliminating at one time. The advantage of this usage will be demonstrated in the segmentation application afterwards.

A detail comparison between the Chinese stop word list generated in our algorithm and the stop word list of Brown corpus [6], which is a well known and widely used

Table 7. Partial of theChinese Stop List and the general English Stop List

Chinese Stop Words	English Stop Words
的(of), 和(and), 在(in), 了(-ed), 一(one), 为(for), 有(have), 中(in/middle), 等(etc.), 是(is), 上(above/on/up), 与(and), 年(year), 对(to), 将(will/shall/would), 到(at/to), 从(from), 不(not), 说(say), 目前(now/nowadays/present), 百分之(percent), 还(also/and), 地(-ly), 并(also/else), 使(cause/make), 他(he), 多(many/more/much), 进行(-ing), 这些(these), 但是(but), 同(and/with), 一个(an/one), 这个(the/this), 之后(after), 下(below/down), 有关(about), 于是(so/therefore/thus), 而(moreover), 但是(but/however), 也(also), 向(to), …	the, of, and, to, a, in, that, is, was, he, for, it, with, as, his, on, be, at, by, I, this, had, not, are, but, from, or, have, an, they, which, you, were, her, all, she, there, would, their, we, him, been, has, when, who, will, …

Table 8. Overlapping Comparison of the Chinese Stop List and the general English Stop List

No. of Stop Words at the Top of List	Overlapping of English and Chinese Stop List
100	81%
200	89%
300	92%

corpus in English ,is done (Table 8). The result shows that the percentage of stop words intersected among two stop word lists is very high, which means that our stop word list in Chinese is comparable with English.

3.3 Stop Word Lists in Segmentation

The importance of word segmentation in Chinese text information retrieval has drawn attention of many researchers. Experiments prove that the effect of segmentation on retrieval performance is ineluctable. Better recognition of a higher number of words generally contributes to the improvement of Chinese information retrieval effectiveness.

As we mentioned, numerous methods have been proposed for segmentation, while none of them took into consideration of Chinese stop word lists. Either because of no standard list is available up till now, or nobody pays attention to these little words. Study on Chinese segmentation found that a large percentage of segmentation errors

come from common single-character words, such as "的"(of), "是"(be), and "个"(ge) [4]. Therefore, we implemented a Chinese segmentation method using our stop word list to demonstrate its effectiveness.

We implemented the segmentation method by boundary detection, which is the same as the segmentation method we used in the construction section. The major modification is that while calculating the bi-grams and tri-grams mutual information, we will multiply the final values with factor 0.5, if any proper substring of exists in the stop word list. On the opposite, the values will be multiplied by a factor of 1.5 if the whole string is matched with entries in the stop word list. The reason for this modification is mainly because of the ambiguity of Chinese words or characters. Our motivation is to avid wrong elimination.

The mutual information equations will be modified as following:

$$\begin{cases} I'(a,b) = I(a,b) \times 0.5 & \text{if proper substring of } S \in \text{Stop List} \\ I'(a,b) = I(a,b) \times 1.5 & \text{if } S \in \text{Stop List} \\ I'(a,b) = I(a,b) & \text{Otherwise} \end{cases} \tag{7}$$

This approach purposes to detect the segmentation points. If any proper substring of a bi-gram or tri-gram appears to be a stop word means that it might be quite possible that it is not a word, so that the value of the point should be reduced and less than original.

中国 改革 和 发展的全局继续保持了稳定

(The progress of reformation and development of China keeps stable)

Without stop words:
[中国] [改革和发展的] [全局] [继续] [保持了] [稳定]

With stop words:
[中国] [改革] [和] [发展] [的] [全局] [继续] [保持] [了] [稳定]

Fig. 1. Comparison of segmentation results with and without using stop words.

In order to measure the accuracy of the segmentation result, we have measure the precision and recall of our segmented text against manually segmented texts. These training texts and test data are taken from a recent international Chinese segmentation competition. A sentence is taken as an example to show the difference of the segmentation results of our method and the original segmentation method (Figure1). Differences occur in the identification of words like "的" (of), "和" (and) and "了" (-ed). With the help of the stop word list, we could figure out those tiny words and segment correctly. In our experiment, the segmentation recall and precision is greatly improved from original 65.3% and 71.1% to 95.24% and 90.1% respectively. The competition has reported an average precision between 84.2% and 89%, and an average recall between 87.2% and 92.3%. The advantage of our segmentation, compared with other methodology [13], is that we make use of our stop word list in simple segmentation method and improve the performance quite a lot.

4. Conclusion

Chinese stop word lists are indispensable in the research of information retrieval. In the paper, we purpose an automatic algorithm for construction of stop word lists in Chinese and generate a generic Chinese stop word list as well. Comparison between our Chinese stop word list and a standard stop word list in English shows that the percentage of stop words intersection is very high, which verifies the effectiveness of our model. Extensive experiments have been conducted on Chinese segmentation to investigate the effectiveness of the stop word list extracted. The results showed that the stop word list can improve the accuracy of Chinese segmentation significantly. Our stop word extraction algorithm is a promising technique, which saves the time for manual generation. It could be applied into other languages in the future.

References

1. Kuang-hua Chen, Hsin-Hsi Chen, Cross Language Chinese Text Retrieval in NTCIR Workshop: towards Cross Language multilingual Text Retrieval, ACM SIGIR Forum, Volume 35 , Issue 2, 2001,pp.12-19.
2. Lin Du, Yibo Zhang, le Sun, yufang Sun and Jie Han, PM-Based Indexing for Chinese Text Retrieval, *Proceedings of the Fifth International Workshop on Information Retrieval with Asian Languages.*
3. Schubert Foo, Hui Li, Chinese word segmentation and its effect on information retrieval, *Information Processing and Management: an International Journal*, v.40 n.1, p.161-190, January 2004.
4. Xianping Ge,Wanda Pratt, Padhraic Smyth, Discovering Chinese words from unsegmented text, In *SIGIR-99* (Proceedings on the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, August 15-19, 1999, Berkeley, CA USA), pages 271-272.
5. Paul B. Kantor, Jung Jin Lee, The maximum entropy principle in information retrieval, Annual ACM Conference on Research and Development in Information Retrieval *Proceedings of the 9th annual international ACM SIGIR conference on Research and development in information retrieval*, page269-274, 1986.
6. H. Kucera, W. Francis, Computational analysis of present day American English, Providence, *RI: Brown University Press*, 1967.
7. I.M.Liu, Descriptive-unit analysis of sentences: Toward a model natural language processing, *Computer Processing of Chinese Oriental Languages*, Vol. 4, No.4, 1990, pp.314-355.
8. K.T.Lua, and G.W. Gan, An application of information theory in Chinese word segmentation, *Computer Processing of Chinese & Oriental Languages*, Vol. 8, No.1, 1994, pp.115-124.
9. Hiroshi Nakagawa, Hiroyuki Kojima and Akira Maeda, Automatic Term Extraction Based on Perplexity of Compound Words, *IJCNLP 2005*, pp. 269-279.
10. Baeza-Yates Ricardo and Ribeiro-Neto Berthier, Modern Information Retrieval, *Addison Wesley Longman Publishing Co. Inc.*
11. J.van Rijsbergen, Information Retrieval, Second Edition, Department of Computer Science, *University of Glasgow, Butterworths, London,* 1979.
12. Roger B. Myerson, Fundamentals of social choice theory, Discussion Paper No. 1162, September,1996

13. Wei-Yun Ma, Keh-Jiann Chen, Introduction to CKIP Chinese Word Segmentation System for the First International Chinese Word Segmentation Bakeoff, *Proceedings of ACL, Second SIGHAN Workshop on Chinese Language Processing*, pp. 31–38.
14. E.Shannon, mathematical theory of communication, *Bell System Technical Journal*, vol. 27, July and October, 1948, pp. 379--423 and 623--656.
15. Sproat, R., Emerson, T.: The First International Chinese Word Segmentation Bakeoff, *The Second SIGHAN Workshop on Chinese Language Processing*, Sapporo, Japan, July 2003.
16. Sproat, and C.L.Shih, A statistical method for finding word boundaries in Chinese text, *Computer Processing of Chinese & Oriental Languages*, vol.4, no. 4, 1990, pp. 336-351.
17. Salton, G., Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*, 24, (1988), 513-523.
18. Zimin Wu and Tseng Gwyneth, Chinese text segmentation for text retrieval achievements and problems, *Journal of the American Society for Information Science*, Vol. 44, No.9, 1993, pp.531-542.
19. Christopher C. Yang, JohnnyW.K. Luk, Stanley K. Yung, and Jerome Yen, Combination and Boundary Detection Approaches on Chinese Indexing, *Journal of the American Society for Information Science*, Vol. 51, No.4, 2000, pp.340-351.
20. Yiming Yang, Noise Reduction in a Statistical Approach to Text Categorization, Proceedings of SIGIR-95, *18th ACM International Conference on Research and Development in Information Retrieval*.
21. K. Zipf, Selective Studies and the Principle of Relative Frequency in Language, Cambridge, *MA: MIT Press*, 1932 .
22. DTIC-DROLS English Stop Word List http://dvl.dtic.mil/stop_list.html
23. An English stop word list in WordNet www.d.umn.edu/~tpederse/Group01/WordNet/words.txt

Fast Search Algorithm for Sequences of Hierarchically Structured Data

Masaki Murata[1], Masao Utiyama[1], Toshiyuki Kanamaru[1,2], and Hitoshi Isahara[1]

[1] National Institute of Information and Communications Technology,
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0289, Japan,
{murata,mutiyama,isahara}@nict.go.jp,
WWW home page: http://www.nict.go.jp/jt/a132/members/murata/
[2] Kyoto University,
Yoshida-nihonmatsu-cho, Sakyo-ku, Kyoto, 606-8501, Japan,
kanamaru@hi.h.kyoto-u.ac.jp

Abstract. We developed an algorithm for quickly searching sequences of hierarchically structured data, such as tagged corpora where each word includes information on its part of speech (POS) and minor POS and the word itself. Using our method, we first make a data item where each data item in a lower level is surrounded by two data items in a higher level. We then connect these data items to make a long string and store the string in a database. We use suffix arrays to query the database. Our experiments showed that our method was 194 times faster than a conventional method at fastest and 24 times faster on average. Our method can be used for other kinds of hierarchically structured data, such as Web applications. Methods that can be used on such data are in high demand. For example, our method can be used to retrieve Web text that includes hierarchical information of low, middle, and high semantic levels. If we use our method for such Web text, we can query using the terms, "High semantic level: method", "Word: in", and "Low semantic level: group"; in other words, our retrieval method is more useful and convenient than conventional Web retrieval.

1 Introduction

We developed an algorithm for quickly searching sequences of hierarchically structured data.[3] For example, in the Kyoto Text Corpus [1], each word has information on its part of speech (POS) and minor POS and the word itself. (A minor POS is a more specific POS.) The POS, minor POS, and word can be considered data of the highest layer, the second-highest layer, and the lowest layer. Hierarchically structured data, as explained below, has data from the lowest to the highest layers. The Kyoto Text Corpus has such a structure, so it can be considered to be sequences of hierarchically structured data. The algorithm we propose is for quickly searching sequences of such data. Our algorithm

[3] We obtained a Japanese patent for the algorithm.

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 163-174

Table 1. Example of sequences of hierarchically structured data.

Word	Minor POS	POS
The	definite article	article
technology	common noun	noun
in	preposition	particle
Japan	proper noun	noun
is	copula	verb
...	...	...

can be used not only for the Kyoto Text Corpus, but also for any sequences of hierarchically structured data.

An application of our algorithm is to search sequences of hierarchically structured data, such as a tagged corpus where each word includes information on its POS and minor POS and the word itself, as mentioned above. Our algorithm is useful for quickly searching a natural language processing system that uses such a tagged corpus. Linguists or users (i.e., learners) that would like to use linguistic information would want to quickly search the database using grammatical information such as POSs as mentioned above. Our algorithm is also useful for this case. Our method can be used for other kinds of hierarchically structured data, such as Web applications. Methods that can be used on such data are in high demand. For example, our method can be used to retrieve Web text that includes hierarchical information of low, middle, and high semantic levels. If we use our method for such Web text, we can query using the terms, "Higher semantic level: method", "Word: in", and "Low semantic level: group"; in other words, our retrieval method is more useful and convenient than conventional Web retrieval.

2 Algorithm

2.1 How to store data in a database

In our algorithm, we use suffix arrays [2] for retrieval. We can perform fast searches by storing data in a database in a special form.

Data are stored in the following form. In our algorithm, we first create a data item where each data item in a lower level is surrounded by a pair of higher-level data items. We then connect these data items to make a long string and store the string in a database. We make indexes for fast searches by using suffix arrays.

For example, assuming that we were going to store the sentence in Table 1 in a database, we would first transform the word "The" into the appropriate form for the database. For this data item, the lowest level data is "The", and the second lowest data level is "definite article", so we put "definite article" on both sides of "The" and obtain the expression "definite article:The:definite article". We use ":" as a boundary between the data items. The third lowest data item is "article", so we put "article" on both sides of "definite article:The:definite article" and

obtain the expression, "article:definite article:The:definite article:article". This is the complete expression for the data item "The" stored in the database. We transform the words "technology", "in", "Japan", and "is" in the same way and obtain the following string.

```
/article:definite article:The:definite article:article/
noun:common noun:technology:common noun:noun/
particle:preposition:in:preposition:particle/
noun:proper noun:Japan:proper noun:noun/
verb:copula:is:copula:verb/
...
```

We store this string in a database with a "/" between expressions. Above, the string is separated into words, and each expression is on a separate line for easy viewing. However, in the database, we do not separate the expressions. Instead, we connect them into strings.

2.2 Method of retrieval

When we query the database, we transform the query into a string as explained above because the database is constructed from hierarchically structured data. However, we transform data items at the beginning and end of a query in different ways. Data items at the beginning of a query are put in order from the lowest layer to highest. Data items at the end of a query are put in order from the highest layer to lowest.

For example, assume that we make the following query.

```
common noun, noun
in, preposition, particle
proper noun, noun
```

That is, the POS of the first word in the query is noun, and its minor POS is common noun; the second word is "in", its POS is particle, and its minor POS is preposition; the POS of the third word is noun, and its minor POS is proper noun.

In this case, the data item at the beginning of the query is transformed to "common noun:noun/" in order from the lowest layer to highest. The data item in the middle of the query is transformed to "particle:preposition:in:preposition: particle/", where each data item in a lower level is surrounded by a pair of higher-level data items. The data item at the end of the query is transformed to "noun:proper noun" in order from the highest layer to lowest. As a result, we construct the following string from the query.

```
common noun:noun/
particle:preposition:in:preposition:particle/
noun:proper noun
```

We search for this string in a database by using a suffix array.

Suffix arrays are known broadly as algorithms for quickly searching for a substring in a string. The algorithm has $log(n)$ steps when the length of the string is n.

Our algorithm can search for a complicated hierarchically structured datum by searching only once using a suffix array.

Although we showed the case where the query is a sequence of three data items (words) and three layers are used for a hierarchical structure, our method can handle sequences of more than three items and more than three layers.

Now, we assume that we query the following.

common noun, noun
particle

That is, the first word's POS is noun, its minor POS is common noun, and the second word's POS is particle. In this case, the data item for the first word is transformed into "common noun:noun" in order from the lowest layer to highest. The data item for the second word is transformed into "particle" in order from the highest layer to lowest. The transformed query is as follows:

common noun:noun/particle

In this case, too, our algorithm can search for a hierarchically structured datum by searching only once using a suffix array.

2.3 Supplement

In our algorithm, higher-level data items are put on both sides of lower-level data items. If we adopt another method where higher-level data items are put on one side of lower-level data items, our algorithm cannot be used to search. For example, when we adopt the wrong method, the data in Table 1 is stored as follows.

/The:definite article:article/
technology:common noun:noun/
in:preposition:particle/
Japan:proper noun:noun/
is:copula:verb/
...

In this case, the query that the first word's POS is noun, its minor POS is common noun, and the second word's POS is particle as in the following cannot be retrieved using a single search with a suffix array.

common noun:noun
particle

This is because an obstructive string, "in:preposition:", is between "common noun:noun" and "particle", and a query cannot be expressed with an unsplit string. Instead, in our algorithm, the data sequence is stored as follows.

/article:definite article:The:definite article:article/
noun:common noun:technology:common noun:noun/
particle:preposition:in:preposition:particle/
noun:proper noun:Japan:proper noun:noun/
verb:copula:is:copula:verb/
...

In this case, what separates "common noun:noun" and "particle" is a separation symbol, "/", only. Therefore, when we would like to search the following query, we make the transformed query, "common noun:noun/particle".

common noun:noun
particle

We can make the query by searching only once using a suffix array.

2.4 Notice for the algorithm

In our algorithm, a data sequence that can be searched for using a one-time retrieval must fulfill the following conditions: The data items in the middle of the query must have data for all the layers, from the lowest to highest, and the data items at the beginning or end of the query must have data for all the layers that are higher than a certain layer. A data sequence must consist of consecutive items, so in a data sequence, there must not be gaps. A data sequence that does not satisfy the above conditions cannot be expressed with a connective string and cannot be searched for using one-time retrieval.

For example, the query specifying that the first word's minor POS is common noun and the second word's POS is particle, as follows, cannot be searched for using one-time retrieval.

common noun
particle

A string in the database is as follows.

noun:common noun:technology:common noun:noun/
particle:preposition:in:preposition:particle/

The word "noun" is between "common noun" and "particle". Therefore, the query cannot be expressed with a connective string and cannot be searched for using one-time retrieval.

To solve the problem, we should use our knowledge of hierarchically structured data and complement data items in higher layers before making the query. For example, the data item in the layer above "common noun" is "noun". Therefore, when the query is the following,

common noun
particle,

Table 2. Average retrieval time.

Retrieval time in conventional method	Retrieval time in our method	Ratio of retrieval time
0.410 s	0.017 s	24.3

"noun" is the complement of "common noun"[4], so we assume that the query is the following.

common noun, noun
particle

The transformed query is "common noun:noun/particle", and we can make the query by using a one-time retrieval.

3 Experiments

Our algorithm is very fast because it uses suffix arrays, a fast search algorithm, only once. We experimented to determine by how much our algorithm is faster than a conventional method. In the experiments, we used our algorithm and the following conventional method.

In the conventional method, we first divide a query into two parts. We next search for the first part of the query using a suffix array. We then check whether each result of the first search has the second part of the query and output the results having the second part of the query as the final retrieval results.

We used Sun UE420R (450 MHz) systems as the computers for the experiments. We used C to program them.

We used the Kyoto Text Corpus Version 2.0 [1] (which has about 20,000 Japanese tagged sentences) as a database. We used the POS information as the data in the highest layer, the minor POS information as the data in the middle layer, and the word information as the data in the lowest layer. For example, in the data item "technology, common noun, noun", "noun" is treated as the data item in the highest layer, "common noun" is treated as the data item in the middle layer, and "technology" is treated as the data item in the lowest layer.

The experimental results are shown in Tables 2 to 4. In the experiments, we used the 24 expressions shown in the first column of Table 3 as the first parts of our queries and used the 51 expressions shown in the first column of Table 4 as the second parts. We made 1224 (= 24 × 51) queries by making all the combinations of the first and second parts and used the queries for the experiments. In our method, we connect the first and second parts of a query

[4] In some cases, a data item in a lower layer has plural data items in a higher layer. In such a case, we have to make queries using all the data items and make OR retrieval using the queries, have the user select one of the data items and make the retrieval using the query made using it, or use word sense disambiguation techniques to select an appropriate data item and make the retrieval using a query made using it.

Table 3. Retrieval time for first part of query.

First part of query	Number of items output	Time (s) (conventional)	Time (s) (our method)	Ratio of time
verb	48692	1.123	0.017	65.1
adjective	11213	0.274	0.018	15.5
prefix	3150	0.082	0.017	4.7
adnoun	532	0.025	0.016	1.5
pronoun	4513	0.120	0.017	7.1
noun	178487	3.955	0.020	194.0
verbal noun:noun	45110	1.103	0.019	58.6
common noun:noun	87130	2.085	0.017	120.8
formal noun:noun	4704	0.130	0.015	8.5
person name:noun	6845	0.172	0.015	11.7
location name:noun	10196	0.257	0.016	16.0
organization name:noun	3249	0.091	0.019	4.9
proper noun:noun	97	0.020	0.018	1.1
prime minister:common noun:noun	447	0.030	0.018	1.7
goverment:common noun:noun	529	0.033	0.019	1.7
expectation:verbal noun:noun	152	0.018	0.016	1.1
hito (human):common noun:noun	542	0.030	0.015	2.0
Murayama:person name:noun	232	0.021	0.016	1.3
Tokyo:location name:noun	390	0.023	0.014	1.7
mono (thing):formal noun:noun	681	0.035	0.017	2.0
koto (matter):formal noun:noun	2255	0.088	0.019	4.7
no (thing):formal noun: noun	1350	0.061	0.015	4.1
omou (think)::verb	120	0.020	0.015	1.4
aru (exist)::verb	1228	0.054	0.018	3.0

into one string and search for it in a database. Using the conventional method, we first search for the first part of a query in a database and check whether each result of the first search has the second part of the query. We then output the results having the second part of the query as the final retrieval results.

Table 2 shows the average retrieval times for the conventional method and our method and the ratio of the average retrieval time of the conventional method to that of our method. Our method is very fast (0.017 s), and the conventional method is slower (0.410 s). Our method is 24 times faster than the conventional method. This indicates that our method is effective.

Table 3 shows the average retrieval time for the queries where the first part of the query is the corresponding first part in the first column of the table and the second part is any of the second parts from Table 4. Table 4 shows the average retrieval time for the queries where the second part of the query is the corresponding second part in the first column of the table and the first part is any of the first parts from Table 3. In the tables, "Number of items output" indicates the number of items output using only the first or second part of the query. (That is, "Number of items output" indicates the number of expressions matching the first or second part of the query.) "Time (conventional)" indicates the average retrieval time for the conventional method. "Time (our method)" indicates the average retrieval time for our method. "Ratio of time" indicates

Table 4. Retrieval time for second part of query.

Second part of query	Number of items output	Time (s) (conventional)	Time (s) (our method)	Ratio of time
verb	48692	0.396	0.015	27.2
aux. verb	4074	0.417	0.018	23.3
adjective	11213	0.406	0.015	26.4
postfix	37322	0.420	0.020	20.6
copula	4616	0.403	0.018	22.5
noun	178487	0.408	0.020	20.4
noun:verbal noun	45110	0.400	0.016	24.6
noun:common noun	87130	0.425	0.018	24.3
noun:formal noun	4704	0.407	0.015	27.1
noun:person name	6845	0.425	0.014	30.9
noun:location noun	10196	0.405	0.017	23.7
noun:organization noun	3249	0.420	0.017	25.2
noun:proper noun	97	0.392	0.018	21.4
particle	125877	0.417	0.022	18.5
particle:case particle	68951	0.410	0.018	22.3
particle:postfix particle	860	0.422	0.015	27.4
particle:sub particle	24565	0.394	0.019	21.0
particle:connective particle	31501	0.410	0.020	20.9
particle:case particle:*kara* (from)	2286	0.419	0.017	24.5
particle:case particle:*ga* (sub.)	12240	0.404	0.014	28.5
particle:case particle:*de* (in)	6901	0.400	0.014	29.1
particle:case particle:*to* (with)	11248	0.402	0.021	18.9
particle:case particle:*ni* (to)	15697	0.416	0.018	23.8
particle:case particle:*no* (sub.)	1601	0.421	0.018	24.1
particle:case particle:*ha* (sub.)	1	0.394	0.015	25.5
particle:case particle:*he* (to)	883	0.426	0.018	23.3
particle:case particle:*made* (to)	795	0.395	0.016	24.9
particle:case particle:*yori* (from)	232	0.426	0.020	20.9
particle:case particle:*wo* (obj.)	17064	0.411	0.016	25.3
particle:postfix particle:*ka*	699	0.407	0.013	31.5
particle:postfix particle:*ne*	47	0.405	0.019	21.6
particle:postfix particle:*yo*	41	0.406	0.019	21.2
particle:postfix particle:*wa*	4	0.407	0.017	24.4
particle:sub particle:*kurai* (about)	19	0.412	0.014	29.9
particle:sub particle:*gurai* (about)	17	0.422	0.022	19.5
particle:sub particle:*koso* (even)	87	0.417	0.016	26.3
particle:sub particle:*sae* (even)	52	0.403	0.015	27.7
particle:sub particle:*shika* (only)	127	0.404	0.015	26.9
particle:sub particle:*sura* (even)	41	0.404	0.015	27.7
particle:sub particle:*dake* (only)	580	0.415	0.016	25.5
particle:sub particle:*datte* (even)	11	0.417	0.016	25.7
particle:sub particle:*demo* (even)	482	0.407	0.017	24.4
particle:sub particle:*nado* (etc.)	1555	0.417	0.018	23.3
particle:sub particle:*nomi* (only)	24	0.412	0.018	23.5
particle:sub particle:*bakari* (only)	49	0.408	0.016	25.1
particle:sub particle:*made* (also)	6	0.408	0.015	26.5
particle:sub particle:*mo* (also)	4663	0.422	0.016	26.6
particle:connective particle:*to* (and)	15	0.417	0.013	32.3
particle:connective particle:*no* (of)	25739	0.400	0.015	25.9
particle:connective particle:*ya* (or)	1530	0.409	0.019	21.8
particle:connective particle:*mo* (too)	7	0.419	0.017	24.5

the ratio of the average retrieval time of the conventional method to that of our method.

From Table 3, we found that when the number of items output using the first part of a query is larger, the time ratio of the conventional method to that of our method is also larger. That is, when the number of items output using the first part of a query is large, the conventional method needs much more time than our method for retrieval. For example, when the first part of a query is "noun", the number of items output is very large (178487). Therefore, the conventional method needs 4 s. On the other hand, our method is very fast (0.020 s). The ratio of time is 194, so our method is 194 times faster than the conventional method. This indicates that our method is effective.

On the other hand, when the number of items output using the first part of a query is small, the ratio of time is not so large. For example, when the first part of a query is "proper noun:noun", the number of items output is small (97). In this case, the retrieval times of the conventional method and our method are 0.020 and 0.018 s and are very similar.

Next, let us examine the results for the second parts of the queries using Table 4. In this case, the conventional method needs almost the same time for each query. We found that the retrieval time of the conventional method does not depend on the second part of a query.

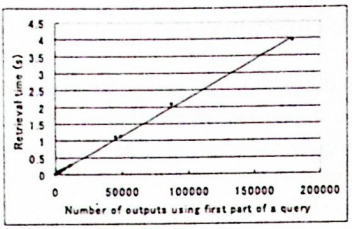
For a more detailed examination, we used the results in Tables 3 and 4 to graph the relationships between the number of items output and retrieval time of the conventional method, between the number of items output and retrieval time of our method, and between the retrieval time of the conventional method and that of our method. The graphs are shown in Figure 1.

Using Figure 1(a1), we found that the retrieval time of the conventional method is roughly proportional to the number of items output using the first part of a query. The reason is that the conventional method has to check whether each result in the first search has the second part of the query.

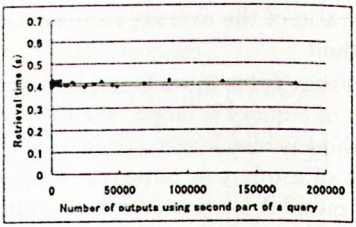
Next, using Figure 1(a2), we can see that the retrieval time of the conventional method does not depend on the number of items output using the second part of a query. The reason is that the conventional method has to check whether each result in the first search has the second part of the query, and the retrieval time does not depend on the number of data items in a database corresponding to the second part of the query.

Using Figures 1(b1) and 1(b2), we can see that the retrieval time of our method is slightly longer when the number of items output using the first or second part of a query is larger.

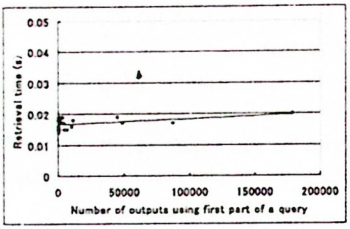
Next, we examined the ratio of the retrieval time of the conventional method to that of our method using Figures 1(c1) and 1(c2). Because the ratio is the retrieval time of the conventional method divided by that of our method, the values in Figures 1(a1) and 1(a2) divided by the values in Figures 1(b1) and 1(b2) are the values in Figures 1(c1) and 1(c2). Because the values in Figure 1(a1), which form a straight line from the lower left side to the upper right side, are divided by the values in Figure 1(b1), which form a straight line tilted down



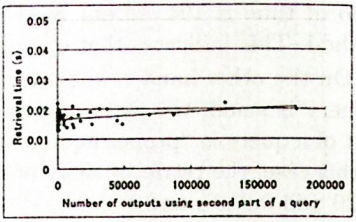
(a1) Time of conventional method.



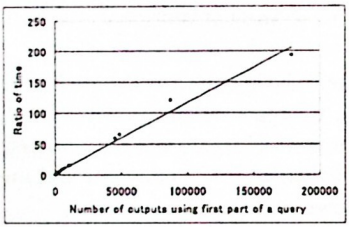
(a2) Time of conventional method.



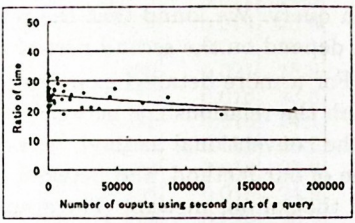
(b1) Time of our method.



(b2) Time of our method.



(c1) Ratio of time.



(c2) Ratio of time.

Fig. 1. Retrieval times for first or second part of query.

slightly on the left, the values in Figure 1(c1) go up to the right but curve down slightly compared to a straight line. Because the values in Figure 1(a2), which form a straight line from left to right, are divided by the values in Figure 1(b2), which form a line going down slightly to the left, the values in Figure 1(c2) go down slightly to the right.

From these results, we found that our method is especially effective when the number of items output using the first part of a retrieval is large. We also found that when the number of items output using the second part of a retrieval is small, the speed of our method is slightly reduced, and the difference between the speeds of our method and the conventional method is slightly smaller.

In the conventional method described above, we always used the first part of a query first and checked whether each result of the first search had the second part of the query to more easily analyze experimental results. However, in general, we first determine whether the first or second part of a query has the smaller

Table 5. Average retrieval time

Retrieval time of conventional method 2	Retrieval time of our method	Ratio of retrieval time
0.112 s	0.017 s	6.65

Table 6. Example of data sequence that includes semantic information.

Word	Semantic Level 2	Semantic Level 1
The	None	None
technology	Method	Abstraction
in	None	None
Japan	Country	Group
is	None	None
...	...	...

number of items to output, then use the part with the smaller number for the first retrieval, and check whether each result in the retrieval has the other part of the query. This is because when the number of the outputs in the first retrieval is smaller, the conventional method is faster. Therefore, we experimented using this method. We call it *conventional method 2*. The results are shown in Table 5. The retrieval time of conventional method 2 is 0.112 s, which is faster than the conventional method (0.410 s). However, the retrieval time of conventional method 2 is 6.2 times slower than that of our method.

We can use SQL to handle hierarchically structured data, so we experimented using it. However, when we used MySQL, we needed a few seconds to search for only one word[5], and we needed a few minutes to a few hours to search for a combination of two words[6]. Therefore, our method is much faster at retrieving one word (100 times faster than the method using SQL; our method needs 0.02 s, and the method using SQL needs a few seconds) and even faster at retrieving a pair of words (6000 to 360000 times faster than the method using SQL; our method needs 0.02 s, and the method using SQL needs a few minutes or a few hours) than the method using SQL.

4 Application

Our algorithm can be used not only for the Kyoto Text Corpus, but also for any sequences of hierarchically structured data.

For example, our method can be used for data sequences using the semantic information of a hierarchical thesaurus such as in Table 6. In this case, the data sequence in the table is transformed into the following string to store it in a database.

[5] The retrieval time for "noun" was 3.59 s.

[6] The retrieval time of the pair "noun:common noun" and "*de* (in):particle" was 3.5 hours.

/None:None:The:None:None/
Abstraction:Method:technology:Method:Abstraction/
None:None:in:None:None/
Group:Country:Japan:Country:Group/
None:None:is:None:None/
...

In the data, we can retrieve a query using the terms, "Semantic level 2: Method", "Word: in", and "Semantic level 1: Group"; that is, we can retrieve the semantic query of "Method in Group". In this case, we complement the query with some data items using our knowledge of hierarchical thesauruses such as WordNet [3] and make the string "Method: Abstraction/None:None:in:None:None/Group". By using this string, we can make the query using a one-time retrieval.

5 Conclusion

We developed an algorithm for quickly searching sequences of hierarchically structured data, such as a tagged corpus where each word includes information on its part of speech (POS) and minor POS and the word itself. To determine the effectiveness of our method, we experimented to compare the retrieval time of our method with that of a conventional method. The conventional method was the method dividing the query into two parts, making the first retrieval using one of the two parts, and checking whether each of the results in the first retrieval included the other part or not. The experimental results showed that our method was 194 times faster than the conventional method at fastest and 24 times faster on average. We also found that our method is especially effective when the number of items output using the first retrieval is large.

Our algorithm can be used for other kinds of sequences of hierarchically structured data. We showed sequences of hierarchically structured data where each word has a higher semantic label and a lower semantic label and described how to transform a sequence of such data into a string to store the data in a database.

Our technique for putting semantic information into text can be used for Web text. Our method can therefore be used for Web retrieval, which is in high demand. If we use our method for Web text, we can query using the terms, "Semantic level 2: Method", "Word: in", and "Semantic level 1: Group"; in other words, our retrieval method is more useful and convenient than conventional Web retrieval.

References

1. Sadao Kurohashi. Kyoto Text Corpus Version 2.0. 1998. (in Japanese).
2. U. Manber and G. Myers. Suffix Arrays: a newmethod for on-line string searches. *SIAM Journal on Computing*, 22(5):935–948, 1993.
3. Princeton University. WordNet 2.0. 2003.

Question Answering and Text Summarization

Cross-Lingual Question Answering Using Inter Lingual Index Module of EuroWordNet*

Sergio Ferrández and Antonio Ferrández

Natural Language Processing and Information Systems Group
Department of Software and Computing Systems
University of Alicante, Spain
{sferrandez,antonio}@dlsi.ua.es

Abstract. This paper outlines the BRILI cross-lingual English-Spanish-Catalan Question Answering (QA) system. The BRILI is being designed at University of Alicante and will be capable to answer English, Spanish and Catalan questions from English, Spanish and Catalan documents. The starting point is our monolingual Spanish QA system [11] which was presented at the 2005 edition of the Cross-Language Evaluation Forum (CLEF). We describe the extensions to our monolingual QA system that are required, especially the language identification module and the strategy used for the question processing module. The Inter Lingual Index (ILI) Module of EuroWordNet (EWN) is used by the question processing modules. The aim of this is to reduce the negative effect of question translation on the overall accuracy of QA systems.

1 Introduction

The aim of a Question Answering (QA) system is to localize the correct answer to a question in natural language in a non-structured collection of documents, also the situations where the system is not to able to provide an answer should be detected. In the case of a Cross-Lingual QA (CL-QA) system, the question is formulated in a language different from the one of the documents, which increases the difficulty. Nowadays, multilingual QA systems have been recognized as an important issue for the future of Information Retrieval (IR).

In this paper we present BRILI (Spanish acronym for "Question Answering using Inter Lingual Index Module"). It is a CL-QA system for Spanish, English and Catalan. It is designed to localize answers from documents, where both answers and documents are written in the three languages. The system is based on complex pattern matching using NLP tools [1, 4, 7, 12]. Beside, Word Sense Disambiguation (WSD) is applied to improve the system (a new proposal of WSD for nouns based on [2]).

BRILI is fully automatic, including the modules of language identification and question processing. The main goal of this paper is to describe these modules

* This research has been partially funded by the Spanish Government under project CICyT number TIC2003-07158-C04-01 and by the Valencia Government under project number GV04B-268.

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 177-182

and the use of the ILI Module of EuroWordNet (EWN) in order to reduce the negative effects of question translation on the overall accuracy of QA systems.

The rest of this paper is organized as follows: section 2 describes effects of question translation on the precision of QA systems. Afterwards, the architecture of the system is shown and discussed in section 3 (specially the modules of language identification and question processing) and finally, section 4 details our conclusions and future work.

2 Negative effects of question translation

Nowadays, most of the implementations of CL-QA systems are based on four different approaches. The first one uses a translation system to translate question into the language in which the documents are written (this technique is the most used). On the other hand, other systems base their implementations on cross-lingual IR (CL-IR) systems. These implementations use the language of the question to generate queries to a CL-IR system. Besides, some systems translate all the documents into the language of the question. Finally, there are sophisticated implementations [6] where English is used as pivot language.

The low quality of machine translation provides results worse than those obtained in the monolingual task.

The precision of a CL-QA system is mainly affected by a correct translation and analysis of the questions that are received as input. An imperfect translation of the question causes a negative impact on the overall accuracy of the systems [3, 5, 6, 10, 13]. As Moldovan [9] stated, Question Analysis phase is responsible for 36.4% of the total of number of errors in open-domain QA.

For CL-QA, translations are often inexact and quite fuzzy, this fact causes an important decrease on the precision of the systems. For instance, on-line Machines Translation systems generate errors such as translations of names that should be left untranslated. The impact of this kind of mistakes should be controlled and valued.

The number of correct answers is always lower for CL-QA. Usually, the precision on cross-lingual task is approximately 50% lower than for monolingual task [14].

3 Architecture Overview

The starting point of BRILI is our monolingual Spanish QA system, called AliQAn [11], which was presented at the 2005 edition of the Cross-Language Evaluation Forum (CLEF).

AliQAn is based fundamentally on syntactic analysis of the questions and the Spanish documents, where the system tries to localize the answer. In order to make the syntactic analysis, SUPAR [4] system is used, which works in the output of a PoS tagger [1]. SUPAR performs partial syntactic analysis that lets us to identify the different grammatical structures of the sentence. Syntactic blocks (SB) are extracted, and they are our basic syntactic unit to define patterns.

Using the output of SUPAR, AliQAn identifies three types of SB: verb phrase (VP), simple nominal phrase (NP) and simple prepositional phrase (PP).

In the step of the extraction of the answer, AliQAn takes the set of retrieved passages by IR-n[7] and tries to extract a concise answer to the question. In order to extract the answer, the following NLP techniques are used:

- *Lexical level.* Grammatical category of answer must be checked according to the type of the question. For example, if we are searching for a *person*, the proposed SB as possible answer has to contain at least a noun.
- *Syntactic level.* Syntactic patterns have been defined. Those let us to look for the answer inside the recovered passages.
- *Semantic level.* Semantic restrictions must be checked. For example, if the type of the question is *city* the possible answer must contain a hyponym of *city* in EuroWordNet. Semantic restrictions are applied according to the type of the questions. Some types are not associated with semantic restrictions, such as *quantity.*

The next example (question 38, *In Workshop CLEF 2003*) shows the used pattern and the behavior the extraction of the answer:

- [SOL[PP, sp: NP1]] [...] [VP][...] [NP2]

First, NP2 (or PP2) and VP are searched by the system, afterward the NP1 with the answer must be found. Next example shows the process:

- **Question:** ¿Qué presidente de Corea del Norte murió a los 82 años de edad? (What North Korea's president died at the age of 82?)
- **Type:** person
- **List of SB:** [NP, north*korea*president] [VP, to death] [PP, at: age [PP, of: 80]
- **Text:** [...] *Kim Il Sung*, **presidente de Corea del Norte, murió ayer a los 82 años** [...] ([...] *Kim Il Sung*, **president of North Korea, died yesterday at the age of 82** [...])
- **List of SB of sentence:** [...] [NP, kim*il* sung [PP, apposition: president [PP, of: north*korea]]] [VP, to death] [PP, at: age [PP, of: 82] [...]
- **Answer:** Kim Il Sung

BRILI (the architecture is shown in Fig.1) is an automatic and complete system for CL-QA tasks. The system carries out an indexation phase where all the documents are analyzed each one in its own language, in which syntactic and semantic information is stored.

The module of language identification have been developed to automatically distinguish the correct language of the question and documents. It is based on two main techniques: the use of dictionaries (joined dictionaries, specific per-language stopwords) and the use of part-of-word terminology (for example, "ing" in the case of English). This philosophy presents a good precision [8] in Spanish, English and Catalan text.

The phase of Question Analysis is made up of two tasks, the first one consisting on detecting the type of information that the answer has to satisfy to be a candidate of answer (proper name, quantity, date, etcetera), while the second one has as objective selecting the question terms (keywords) that make possible

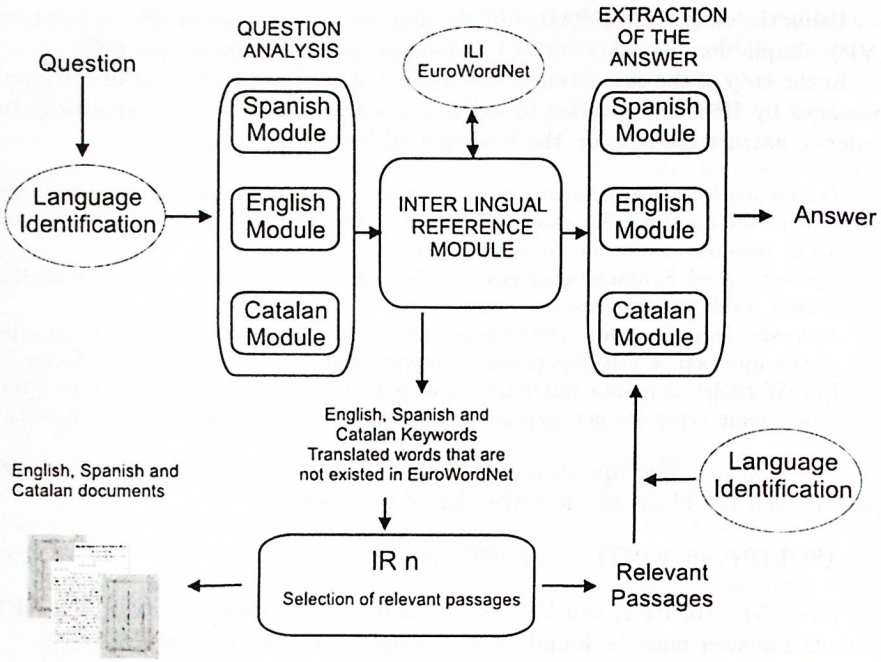


Fig. 1. System architecture

to locate those documents that can contain the answer. The expected answer type is achieved using different sets of syntactic patterns according to the language that is being processed. Beside, WSD is applied to obtain the synset for each keyword.

The next example shows the behavior of question analysis in a question of type *person*:

- Question:
  - *Which French president inaugurated the Eurotunnel?*
- Information used (syntactic blocks) to detect the type of the question and keywords:
  - interrogative pronoun - *Which*
  - nominal phrase - *French president* - synset in English
  - verb phrase - *to inaugurate* - synset in English
  - nominal phrase - *Eurotunnel* - this word does not exit in EuroWoedNet

The inputs of the Inter Lingual Reference (ILR) module are the detected keywords in the question and the type of the question. In addition to the syntactic blocks and the type of the question, the ILR module returns for each keywords its synsets in the three languages, using the ILI Module of EWN. The words that are not in EWN are translated into the rest of the languages using a machine translation system. Our approach does not achieve a translation of the

question, it indexes the words using the ILI of EWN reducing the negative effect of question translation on the overall accuracy. For instance, using the previous example the ILR module selected the synset in Spanish and Catalan for the word *"French"*, on the other hand the word *"Eurotunnel"* which is not presented in EWN is translated into Spanish and English using machine translation.

The behavior of this module is shown using the previous example.

- **Input of ILR module:**
  - *French* synset in English
  - *president* synset in English
  - *to inagurate* synset in English
  - *Eurotunnel* this word does not exit in EuroWoedNet
- **Output of ILR module:**
  - *French* synsets in English, Spanish and Catalan
  - *president* synsets in English, Spanish and Catalan
  - *to inagurate* synsets in English, Spanish and Catalan
  - *Eurotunnel* this word does not exit in EuroWoedNet
    - translations into Spanish and Catalan

The phase of Selection of relevant passages uses IR-n system [7]. The inputs of IR-n are the detected keywords and the translated word that are not in EWN. For instance, using the previous example, IR-n receives as input the words: *"inaugurate"* with its synonymous; *"inaugurar"* (in Spanish) with its synonymous and *"inaugurar"* (in Catalan) with its synonymous. The translated words have been obtained indexing the synsets of words using ILI Module of EWN without using any machine translation. IR-n returns a list of passages where the system applies the extraction of the answer according to the language in which each passage is written.

The final step of BRILI is the phase of Extraction of the Answer which is composed of three monolingual modules. BRILI uses the syntactic blocks of the question and different sets of syntactic patterns (according to the language) with lexical, syntactic and semantic information to find out the correct answer. Next, an example of syntactic pattern for Spanish is shown which captures solution in Spanish sentence.

- **Sentence:**
  "... el Presidente Francés, Jacques Chirac, inauguró el Eurotunel ..." (... the French President, Jacques Chirac, inaugurated the Eurotunnel ...)
- **Syntactic pattern:**
  [NP (*"French president"*), apposition [NP (SOLUTION)]] + [VP (*"to inaugurate"*)] + [NP *"Eurotunnel"*)]

In order to decreased the effect of incorrect translation of the words that are not in EWN, the matches using these words in the search of the answer are valued less than the words obtained from the ILI Module of EWN.

4 Conclusions and Future Work

The main objective pursued by our proposal is a cross-lingual QA system that reduces the use of machine translation. Reducing the decrease of the precision that is caused by the question translation will be achieve by means of the use of the ILI Module of EWN and WSD. Nowadays, BRILI is being implemented. For this reason, results are not presented in this paper.

References

1. S. Acebo, A. Ageno, S. Climent, J. Farreres, L. Padró, R. Placer, H. Rodriguez, M. Taulé, and J. Turno. MACO: Morphological Analyzer Corpus-Oriented. *ESPRIT BRA-7315 Aquilex II, Working Paper 31*, 1994.
2. E. Agirre and G. Rigau. A proposal for word sense disambiguation using conceptual distance. *1st Intl. Conf. on recent Advances in NLP. Bulgaria*, 1995.
3. C. de Pablo-Sánchez, A. González-Ledesma, J.L. Martínez-Fernández, J.M. Guirao, P. Martinez, and A. Moreno. MIRACLE's 2005 Approach to Cross-Lingual Question Answering. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, 2005.
4. A. Ferrández, M. Palomar, and L. Moreno. An Empirical Approach to Spanish Anaphora Resolution. *Machine Translation. Special Issue on Anaphora Resolution In Machine Translation*, 14(3/4):191–216, December 1999.
5. J. M. Gómez, E. Bisbal, D. Buscaldi, and P. Rosso E. Sanchis. Monolingul and Cross-Language QA using a QA-oriented Passage Retrieval System. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
6. D. Laurent, P. Séguéla, and S. Negre. Cross Lingual Question Answering using QRISTAL for CLEF 2005. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
7. F. Llopis and J.L. Vicedo. Ir-n, a passage retrieval system. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, 2001.
8. T. Martínez, E. Noguera, R. Muñoz, and F. Llopis. Web track for CLEF2005 at ALICANTE UNIVERSITY. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
9. D.I. Moldovan, M. Pasca, S.M. Harabagiu, and M. Surdeanu. Performance issues and error analysis in an open-domain question answering system. *ACM Trans. Inf. Syst*, 21:133–154, 2003.
10. G. Neumann and B. Sacaleanu. DFKI's LT-lab at the CLEF 2005 Multiple Language Question Answering Track. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
11. S. Roger, S. Ferrández, A. Ferrández, J. Peral, F. Llopis, A. Aguilar, and D. Tomás. AliQAn, Spanish QA System at CLEF-2005. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, 2005.
12. H. Schmid. TreeTagger — a language independent part-of-speech tagger. *Institut fur Maschinelle Sprachverarbeitung, Universitat Stuttgart*, 1995.
13. R. F. E. Sutcliffe, M. Mulcahy, and I. Gabbay. Cross-Language French-English Question Answering Using the DLT system at CLEF 2005. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.
14. A. Vallin, B. Magnini, D. Giampiccolo, L. Aunimo, C. Ayache, P. Osenova, A. Peas, M. de Rijke, B. Sacaleanu, D. Santos, and R. Sutclife. Overview of the CLEF 2005 Multilingual Question Answering Track. *In Workshop of Cross-Language Evaluation Forum (CLEF)*, September 2005.

Using Terminology and a Concept Hierarchy for Restricted-Domain Question-Answering

Hai Doan-Nguyen and Leila Kosseim

Department of Computer Science and Software Engineering
Concordia University
Montreal, Quebec, H3G 1M8, Canada
haidoan@cse.concordia.ca, kosseim@cse.concordia.ca

Abstract. This paper discusses an important characteristic of restricted-domain question-answering (RDQA): the issue of low precision in finding good answers. We propose methods for improving precision using domain-specific terminology and concept hierarchy to rearrange the candidate list and to better characterize the question-document relevance relationship. Once this relationship has been well established, one can expect to obtain a small set of (almost) all relevant documents for a given question, and use this to guide the information retrieval engine in a two-level search strategy. The methods proposed here have been applied to a real QA system for a large telecommunication company, yielding significant improvements in precision.

1 Introduction

This paper presents our research in the development of a question-answering (QA) system for a restricted domain. The system's goal is to reply to customer's questions on services offered by Bell Canada, a major Canadian telecommunication corporation. Although grounded within a specific context, we try to reveal general problems and develop a general methodology for restricted-domain QA (RDQA).

Although work in RDQA dates back to the early years of Artificial Intelligence, the domain has only recently regained interest in the research community. RDQA performs QA on a specific domain and often uses document collections restricted in subject and volume. Often integrated in real-world applications, RDQA, especially systems working on free text corpora (rather than on structured databases), provides many interesting challenges to natural language engineering. Real questions have no limit on form, style, category, and complexity. In addition, a RDQA system often has to deal with the problem of low precision in finding a correct answer for a given question.

In this paper, we discuss the main characteristics of RDQA, and present a series of methods to improve the precision performance of our system. These methods were not developed specifically for our project at hand but always considered in a general perspective of RDQA.

© *A. Gelbukh (Editor)*
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 183-194

2 Restricted-Domain QA (RDQA)

In the early days of Artificial Intelligence, work in QA typically addressed restricted-domain systems working over databases (e.g., [1] and [2]). However, at the end of the last decade, interest in QA has been stirred up by the availability of huge volumes of electronic documents, especially the WWW, and of powerful search engines. Within this context, researchers have been most attracted to open-domain QA, driven by the TREC and DARPA initiatives. Most work has been performed on finding precise and short answers to factoid questions. However, as QA is applied to more practical tasks, it has become apparent that this orientation cannot satisfy the requirements of practical applications. It is RDQA, re-emerging recently, that brings out a set of new directions to the domain. Expected to carry out QA principally in real-life contexts, particularly in industrial and commercial applications, RDQA has to deal with real and very difficult problems. It is no surprise that RDQA is regaining attention these days at the research level, as evidenced by, e.g., [3–5]. Some typical work on RDQA include the LILOG project that answers questions on tourist information [6], the ExtrAns system working on Unix manuals [7], WEBCOOP also on tourist information [8], and the system of [9] working on the telecommunications domain.

Techniques developed for open-domain QA, particularly for TREC competitions, are not that helpful in RDQA. Indeed, RDQA has several characteristics that make it different from open-domain QA:

Restricted Document Collection In RDQA, the document collection is typically restricted in subject and in volume. By definition, the domain of knowledge and information of interest is predefined, and often very narrow. The working document collection is normally much smaller in size than that of open-domain QA systems. In addition, it is often homogeneous in style and structure.

Scarcity of Answer Sources A consequence of a restricted document collection is the scarcity of answer sources. Redundancy of answer candidates for a given question is exploited extensively in open-domain QA systems as one principal method to determine the right answer. The situation is contrary in RDQA. As the documents often come from only a few sources, they do not likely have repeated contents. Information about a specific issue is typically referred to in only a few areas of the documents, and the system will not have a large retrieval set abundant of good candidates for selection. [10] shows that the performance of a system depends greatly on the redundancy of answer occurrences in the document collection. For example, they estimate that only about 27% of the systems participating in TREC-8 produced a correct answer for questions with exactly one answer occurrence, while about 50% of systems produced a correct answer for questions with 7 answer occurrences. (7 is the average answer occurrences per question in the TREC-8 collection.) Scarcity of correct answers explains why low performance on precision is a general concern for RDQA systems.

Domain Specific Terminology A RDQA system normally has to work with domain-specific terms and meaning. As these will not be found in a general dictionary or thesaurus, lexical and semantic techniques based on the use of these resources (consider the wide use of WordNet in open-domain QA) may not apply well here. Instead we need specialized knowledge, such as a technical term dictionary, an ontology of entities in the domain, etc. Nonetheless, building these resources from scratch may be costly and problematic, because it often requires domain expertise.

On the other hand, as in any specialized domain, word sense disambiguation may be a smaller issue in RDQA. Most "important" words will only have a few possible senses. For instance, the word *card* in our corpus seems to have only one sense (telephone card), compared to 11 senses in WordNet 2.1.

Complex Questions If a QA system is to be used for a real application, e.g. answering questions from clients of a company, it should accept arbitrary questions, of various forms and styles. Contrarily, current open-domain QA systems generally suppose that the questions are constituted by a single, and often simple, sentence, and can be categorized into a well-defined and simple semantic classification (e.g. Person, Time, Location ...). Complexity of questions can be studied at different levels: *representational level*: syntax, number of sentences, style, ...; *semantic level*: entities in the question and their relationships; *logical level*: hypotheses, presuppositions, inferences, ..., and *intentional level:* expectations, objectives, ... For example, let's consider a question from our corpus:

> *It seems that the First Rate Plan is only good if most of my calls are in the evenings or weekends. If so, is there another plan for long distance calls anytime during the day?*

This is a rather complex question at the representational level, because it is made up of two sentences, both, syntactically complex. Semantically, it asserts something about the *First Rate Plan*, and asks for another plan for long distance telephone calls. At the logical level, one can discover that the client has presupposed (correctly) that *First Rate Plan* is a long distance telephone call plan, that this plan is not economical during the day. At the intentional level, she wants to know whether there is another similar but less expensive plan, and expects to receive various information about it (how to register, its price structure ...).

Complex Answers Answers in real-life QA cannot be restricted to simple phrases or short snippets. They should contain as much information as needed in order to answer the true intention of the question. This may imply clarifying the context of the problem posed in the question, explaining different situations, providing justifications and inferences, giving instructions or suggestions, ... For the example above, it would be a business disaster for the company if the QA system only returned a specific plan name, e.g. *"First Rate 24 !"* with no further

explanations or details. Finding an answer that contains the necessary information (*completeness*), but only the necessary information (*conciseness*) is far more difficult with non-factual questions where the answer is not simply defined as a grammatical constituent. An ideal answer may be as follows:

> **A:** *First Rate 24*
> *$4.95/month*
> *If most of your long distance calling is within Canada or to the United States during the day or if you want convenience of calling anytime during the day or night, then the First Rate 24 hours plan may be a good option for you. With the First Rate 24 hours long distance plan you will receive the following benefits.*
> *- Now you never have to worry about the time when you're calling friends, relatives, or following up on an important business call.*
> *- For a low fee of $4.95 a month, you can call anywhere in Canada for just 10 cents a minute, and to the U.S. for just 20 cents a minute, 24 hours a day, everyday.*
> *- You'll also be able to take advantage of our 24 hour flat overseas rates on direct dialled calls to our 28 most popular overseas calling destinations.*

Evaluation Finally, evaluation of a RDQA system cannot be performed using the same techniques as in open-domain QA. In open-domain QA, answers are often very short, hence one can look for a specific pattern in the answers and automate the evaluation process, in order to compute various measures (e.g. MRR, confidence-weighted score, etc). In the case of complex questions and answers, a more complete evaluation is necessary. As an example, [11] present a novel and comprehensive way to evaluate a system as a whole, but approaches inspired by the evaluation of automatic summarization systems can also be used.

3 RDQA for Bell Canada

3.1 The Corpus

In our project, the document collection was derived from Bell Canada's website (`www.bell.ca`) in 2003. It contains descriptions of the company's wide-range products and services in telecommunication (telephone, wireless, Internet, Web, etc.). As the documents were in various formats (e.g. HTML, PDF) they were saved as plain text, sacrificing some important formatting cues. The collection comprises more than 220 documents, for a total of about 560K characters.

The available question set has 140 questions, whose form and style vary freely. Most questions are composed of one sentence, but some are composed of several sentences. The average length of the questions is 11.3 words (to compare, that of TREC questions is 7.3 words). The questions ask about what a service is, its details, whether a service exists for a certain need, how to do something with

a service, etc. For the project, we divided the question set at random into 80 questions for training and 60 for testing. Below are some examples of questions:

> *Do I have a customized domain name even with the Occasional Plan of Business Internet Dial?*
> *With the Web Live Voice service, is it possible that a visitor activates a call to our company from our web pages, but then the call is connected over normal phone line?*
> *It seems that the First Rate Plan is only good if most of my calls are in the evenings or weekends. If so, is there another plan for long distance calls anytime during the day?*

3.2 Initial Evaluation: Using a naïve QA system

Although our collection was not very large, it was not so small either so that a strategy of searching answers directly in the collection could be obvious. Hence we first followed the classic two-step strategy of QA: information retrieval (IR), and then candidate selection and answer extraction. The well-known generic IR engine OKAPI (`www.soi.city.ac.uk/~andym/OKAPI-PACK/`, also [12]) helped us to do both steps. For each question, input as such, OKAPI returns an ordered list of answer candidates, together with a relevance score for each candidate and the name of the document containing it. An answer candidate is a paragraph which OKAPI considers most relevant to the question[1]. OKAPI is thus more than just a traditional IR engine; it can be regarded as a naïve QA system returning paragraphs.

The candidates were then evaluated by a human judge using a binary scale: correct or incorrect. This kind of judgment is recommended in the context of communications between a company and its clients, because the conditions and technical details of a service should be edited as clearly as possible in the reply to the client. However we did also accept some tolerance in the evaluation. If a question is ambiguous, e.g., it asks about phones but does not specify whether it pertains to wired phones or wireless phones, all correct candidates of either case will be accepted. If a candidate is good but incomplete as a reply, it will be judged correct if it contains the principal theme of the supposed answer, and if missing information can be found in paragraphs around the candidate's text in the containing document.

Table 3.2 shows OKAPI's performance on the training question set. We kept at most the 10 best candidates for each question, because after rank 10 a correct answer was very rare. C(n) is the number of all candidates at rank n which are judged correct. Q(n) is the number of questions in the training set which have at least one correct answer among the first n ranks. As for answer redundancy, among the 45 questions having at least a correct answer (see Q(10)), there were

[1] A paragraph is a block of text separated by two newline characters. As formatted files were saved in plain text, original "logical" paragraphs may be joined up into one paragraph, which may affect the precision of the candidates.

33 questions (41.3% of the entire training set) having exactly 1 correct answer, 10 questions (12.5%) having 2, and 2 questions (2.5%) having 3 correct answers. Table 3.2 also gives OKAPI's precision on the test question set.

Table 1. Performance of OKAPI on the training question set (80 questions), and the test question set (60 questions).

n	1	2	3	4	5	6	7	8	9	10
Training Set										
C(n)	20	11	5	4	9	3	1	1	4	1
%C(n)	25%	13.8%	6.3%	5%	11.3%	3.8%	1.3%	1.3%	5%	1.3%
Q(n)	20	26	28	32	39	41	42	43	44	45
%Q(n)	25%	32.5%	35%	40%	48.8%	51.3%	52.5%	53.8%	55%	56.3%
Test Set										
C(n)	18	8	7	2	4	3	3	2	1	1
%C(n)	30%	13.3%	11.7%	3.3%	6.7%	5%	5%	3.3%	1.7%	1.7%
Q(n)	18	23	28	29	32	33	35	36	36	37
%Q(n)	30%	38.3%	46.7%	48.3%	53.3%	55%	58.3%	60%	60%	61.7%

The results show that OKAPI's performance on precision was not satisfying, conforming to our discussion about characteristics of RDQA above. The precision was particularly weak for n's from 1 to 5. Unfortunately, these are cases that the system aims at. n=1 means that only one answer will be returned – a totally automatic system. n=2 to 5 correspond to a more practical scenario of a semi-automatic system, where an agent of the company chooses the best one among the n candidates, edits it, and sends it to the client. We stopped at n=5 because a greater number of candidates seems too heavy psychologically to the human agent. Also note that the rank of the candidates is not considered important here, because they would be equally examined by the agent. This explains why we used Q(n) to measure the precision performance rather than other well-known scoring such as mean reciprocal rank (MRR) or confidence-weighted score.

Examining the correct candidates, we found that they were generally good enough to be sent to the user as an understandable reply. About 25% of them contained superfluous information for the corresponding question, while 15% were lacking information. However, only 2/3 of the latter (that is 10% of all) looked difficult to be completed automatically. Generating a more concise answer from a good candidate (an extracted paragraph) therefore seemed less important than improving the precision of the IR module. We therefore concentrated on how to improve Q(n) for n= 1 to 5.

4 Improving Precision

The first obvious approach to improve the precision performance of the system is to use a better IR engine or tuning the current one to the specific domain and

task, e.g. by adjusting the parameters, modifying the weighting formulas of the engine. However, this approach is neither general (an engine may work well with one application but not with another), nor interesting at the theoretical level.

The second approach consists in improving the results returned by the IR engine. One main direction is re-ranking the candidates, i.e. pushing good candidates in the returned candidate list to the first ranks as much as possible, thus increasing Q(n). To do this, we need some information that can characterize the relevance of a candidate to the corresponding question better than the IR engine did. The most prominent kind of such information may be the domain-specific language used in the working domain of the QA system, particularly its vocabulary, or even more narrowly, its terminological set.

While implementing re-ranking methods, we found that domain-specific semantic information could be used to filter documents according to their relevance to a given question. This led us to another approach: concept-based two-level search. In the following, we will present our development of a series of strategies for precision improvement and their results.

4.1 Re-ranking Candidates

In this approach, we experimented with two methods of re-ranking: one with a strongly specific terminological set, and one with a concept hierarchy allowing a good document characterization.

Re-ranking using terminology We noted that the names of specific Bell services, such as *Business Internet Dial*, *Web Live Voice*, etc., could be used as a relevance characterizing information, because such terms occurred very often in almost every document and question, and a specific service was often presented or mentioned in only one or a few documents, making these terms very discriminating. Luckily, these *domain terms* occur typically in capital letters in the corpus, and could easily be extracted automatically. After a manual filtering, we obtained more than 450 domain terms.

We therefore designed a new scoring method which increases the score of candidates containing occurrences of domain terms found in the corresponding question. Of course, the system also combine Okapi's scoring in the computation. Due to space limit, we refer readers to [13] and [14] for the details of the experiment.

The new terminology-based scoring gave very encouraging improvements on the training set, but just modest results when running with optimal training parameters on the test set (see Table 4.1). What encouraged us here was that improvement was shown to be possible; we just had to look for better characterizing information. The scoring system devised here was also used as the basic model for the next set of experiments.

Re-ranking using a concept hierarchy To better re-rank the candidates, we now focused on how to guess which documents most probably provide a good

Table 2. Best results of the terminology-based scoring on the training set, and results of applying it with optimal training parameters on the test set.
Note: $\Delta Q(n)$ = System's Q(n) - OKAPI's Q(n); $\%\Delta Q(n) = \frac{\Delta Q(n)}{okapi's Q(n)}$.

n	1	2	3	4	5
Training Set					
Q(n)	30	40	42	43	44
$\Delta Q(n)$	10	14	14	11	5
$\%\Delta Q(n)$	50%	53.8%	50%	34.4%	12.8%
Test Set					
Q(n)	22	29	32	33	34
$\Delta Q(n)$	4	6	4	4	2
$\%\Delta Q(n)$	22.2%	26.1%	14.3%	13.8%	6.3%

candidate for a given question. For this purpose, we tried to map the documents into a system of concepts. Each document refers to a set of concepts, and a concept is discussed in a set of documents. Building such a concept system is feasible within closed-domain applications, because the domain of the document collection is pre-defined, the number of documents is in a controlled range, and the documents are often already classified topically, e.g. by their creator. If no such classification existed, one can use techniques of building hierarchies of clusters (e.g. [15]). In our case, a concept hierarchy and the mapping between it and the document collection was easily built based on the original document classification of Bell Canada. Below is a small excerpt from the hierarchy:

```
Bellall
    Personal
        Personal-Phone
            Personal-Phone-LongDistance
                Personal-Phone-LongDistance-BasicRate
                Personal-Phone-LongDistance-FirstRate
```

The use of the concept hierarchy in the QA system was based on the following assumption: *A question can be well understood only when we can recognize the concepts implicit in it.* To be precise, we should measure the relevance of a concept to a given question. For example, the concepts `Personal-Phone-LongDistance-FirstRate` and `Personal-Phone-LongDistance` are highly relevant to the question *"It seems that the First Rate Plan is only good if most of my calls... "* mentioned in section 2. From that measure, it is easy to compute the relevance of a document to a question using the concept-documents mapping. If we keep only the concepts and documents having a positive relevance measure with a question, we have a question-concepts and a question-documents mapping, respectively.

Now it seems that we can better re-arrange the candidates with a new kind of information: a candidate will be ranked higher if its containing document is more relevant to the question. Details of the implementation of this strategy,

including how to measure concept-question and document-question relevance, are given in [14]. The uniformly good improvements (see Table 4.1) show that the approach is appropriate and effective.

Table 3. Best results of the concept-based scoring on the training set, and results of applying it with optimal training parameters on the test set.

n	1	2	3	4	5
Training Set					
Q(n)	32	41	44	44	44
ΔQ(n)	12	15	16	12	5
%ΔQ(n)	60%	57.7%	57.1%	37.5%	12.8%
Test Set					
Q(n)	30	32	35	35	36
ΔQ(n)	12	9	7	6	4
%ΔQ(n)	66.6%	39.1%	25%	20.7%	12.5%

4.2 Concept-Based Two-Level Search

As the concept-based mappings in the previous section seem to be able to point out the documents relevant to a given question with a high precision, we tried to see how to combine it with the IR engine. In the previous experiments, the IR engine searches over the entire document collection. Now search will be carried out in two levels: at the *semantic level*, the system determines a subset of most promising documents for each question; at the *IR level*, the engine just searches in this subset. We hoped that search could achieve higher precision in working with a much smaller document set, which usually contains no more than 20 documents in our case. As we use the concept-based mappings at the semantic level, we will call this strategy *conceptual-mapping-then-IR*.

Conceptual Mapping then Okapi We first experimented our two-level search strategy with OKAPI, where indexing and search were performed for each question on its relevant document subset. The results were not better than when OKAPI worked with the entire document collection. We then applied the scoring system devised in the previous section to rearrange the candidate lists. Although the results on the training set were generally better than those of the previous section, results on the test set were worse, which led to an unfavourable conclusion for this method. Details of the experiments here can be found in [14].

Using a new IR engine The precision of the conceptual mappings was good, but the performance of the two-level system using OKAPI was not convincing. This may be because OKAPI cannot work well with very small document sets.

We therefore implemented another IR engine, which is simple but adapted for working with small document sets. It tries to identify an excerpt in a given document that is most likely to answer the input question. Details of the system is presented in [14]. Note that if the relevant document subset for a given question is empty, the system takes the candidates proposed by OKAPI as results (i.e. *okapi as last resort*).

Also in this implementation, if the searched document is "small", i.e. contains less than 2000 characters, the system simply takes the entire document as the candidate. This reflects the nature of the collection and of our current task: in fact, those small documents are often dedicated to a very specific topic, and it seems necessary to present its contents in its entirety to any related question for reasons of understandability, or because of important additional information in the document. Also, a size of 2000 characters (which are normally 70% of a page) seems acceptable for a human judgment in the scenario of a semi-automatic system. We call this case *"candidate expansion"*, because whatever a candidate may be, it is expanded to the whole document.

The results (Table 4.2) show that except for the case of n=1 in the test set, the new system performs well for precision. What is interesting here is that although simple, the engine is quite effective, because it does searching on a well selected and very small document subset.

Table 4. Best results of the new IR engine on the training set, and results of this method (with optimal training parameters) on the test set.

n	1	2	3	4	5
Training Set					
Q(n)	42	55	60	60	61
ΔQ(n)	22	29	32	28	22
%ΔQ(n)	110%	112%	114%	88%	56%
Test Set					
Q(n)	23	37	41	42	42
ΔQ(n)	5	14	13	13	10
%ΔQ(n)	27.8%	60.9%	46.4%	44.8%	31.3%

Expanding Answer Candidates The previous experiment has shown that extending the size of answer candidates can greatly ease the task. This can be considered as another method belonging to the approach of improving precision by improving the results returned by the IR engine. To be fair, it is necessary to see how precision will be improved if this candidate expansion is used in other experiments. We thus performed two further experiments. In the first one, any candidates returned by OKAPI (cf. Table 3.2) which came from a document of less than 2000 characters were expanded into the entire document. In the second experiment, we similarly expanded candidates returned by *conceptual-*

mapping-then-okapi. Results showed that improvements are not as good as those obtained by other methods. See details in [14]. The two experiments suggest that expanding candidates helps improve the precision, but not so much unless it is combined with other methods. We have not yet, however, carried out experiments of combining candidate expansion with re-ranking.

5 Conclusion and Future Work

RDQA, working on small document collections and restricted subjects, seems to be no less difficult a task than open-domain QA. Due to candidate scarcity, the precision of a RDQA system, and in particular that of its IR module, becomes a problematic issue. It affects seriously the entire success of the system, because if most of the retrieved candidates are incorrect, it is meaningless to apply further techniques of QA to refine the answers.

In this paper, we have discussed several methods to improve the precision of the IR module. They include the use of domain-specific terminology and concept hierarchy to rearrange the candidate list and to better characterize the question-document relevance relationship. Once this relationship has been well established, one can expect to obtain a small set of (almost) all relevant documents for a given question, and use this to guide the IR engine in a two-level search strategy.

Also, long and complex answers are a common characteristic of RDQA systems. Being aware of this, one can design an appropriate system which is more tolerant to answer size to achieve a higher precision, and to avoid the need of expanding a short but insufficient answer into a complete one.

Good improvements achieved when applying our methods to a QA system for Bell Canada shows that these methods are applicable and effective. Although the experiments were grounded in a real-world situation, we have tried to build the strategies as general and as modular as possible in order to develop a general methodology for the task of RDQA.

Many problems on the precision performance of a RDQA system have been suggested in this paper. First, there is the problem of how to build a *good* answer. We have worked with finding correct answers. However, a correct answer may not be *good*, because it may be vague, lengthy, superfluous, etc. The ultimate goal of QA is to find answers that satisfy the questioner's needs rather than just correct ones, and we are currently working on this problem. Second, there is the problem of how to analyze an arbitrary question into semantic and logical parts (entities mentioned in the question and their relationships, pre-suppositions, problem context, question focus, etc.) so that the system has a better understanding of what is being asked. Third, the general problem in Natural Language Processing, of analysing free text at any level. Here, work performed in automatic topic detection and content segmentation may be helpful to identify relevant answers. Certainly, these also constitute problems of open-domain QA if one wants to explore the domain further than the typical factoid questions.

Acknowledgments This project was funded by Bell University Laboratories (BUL) and the Canada Natural Science and Engineering Research Council (NSERC). The authors would also like to thank to anonymous referees for their comments.

References

1. Green, B.F., Wolf, A.K., Chomsky, C., Laughery, K.: Baseball: An Automatic Question Answerer. In: Proceedings of the Western Joint Computer Conference. (1961) 219–224
2. Woods, W.: Progress in Natural Language Understanding: An Application to Lunar Geology. In: AFIPS Conference Proceeding. Volume 42. (1973) 441–450
3. Mollá, D., Vicedo, J.L., eds.: Workshop on Question Answering in Restricted Domains - ACL-2004, Barcelona, Spain (2004)
4. Mollá, D., Vicedo, J.L., eds.: AAAI-05 Workshop on Question Answering in Restricted Domains, Pittsburg, Pennsylvania (2005) to appear.
5. Mollá, D., Vicedo, J.L., eds.: Special Issue of Computational Linguistics on Question Answering in Restricted Domains. (2005) to appear.
6. Herzog, O., Rollinger, C.R., eds.: Text Understanding in LILOG, Integrating Computational Linguistics and Artificial Intelligence, Final Report on the IBM Germany LILOG-Project. In Herzog, O., Rollinger, C.R., eds.: Text Understanding in LILOG. Volume 546 of Lecture Notes in Computer Science., Springer (1991)
7. Mollá, D., Berri, J., Hess, M.: A real world implementation of answer extraction. In: Proceedings of the 9th International Workshop on Database and Expert Systems, Workshop: Natural Language and Information Systems (NLIS-98), Vienna (1998)
8. Benamara, F., Saint-Dizier, P.: Advanced Relaxation for Cooperative Question Answering. In: New Directions in Question Answering. (2004) 263–274
9. Oroumchian, F., Darrudi, E., Ofoghi, B.: Knowledge-Based Question Answering with Human Plausible Reasoning. In: Proceedings of the 5th International Conference on Recent Advances in Soft Computing. (2004)
10. Light, M., Mann, G., Riloff, E., Breck, E.: Analyses for Elucidating Current Question Answering Technology. Natural Language Engineering **7** (2001)
11. Diekema, A.R., Yilmazel, O., Liddy, E.D.: Evaluation of Restricted Domain Question-Answering Systems. In: Proceedings of the Association of Computational Linguistics 2004 Workshop on Question Answering in Restricted Domains (ACL-2004), Barcelona, Spain (2004)
12. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M.: OKAPI at TREC-3. In D.K.Hartman, ed.: Overview of the Third TExt Retrieval Conference (TREC-3), Gaithersburg, NIST (1995) 109–130
13. Doan-Nguyen, H., Kosseim, L.: Improving the Precision of a Closed-Domain Question-Answering System with Semantic Information, Avignon, France, RIAO-2004 (2004)
14. Doan-Nguyen, H., Kosseim, L.: The Problem of Precision in Restricted-Domain Question-Answering. Some Proposed Methods of Improvement, Barcelona, Spain, ACL (2004)
15. Kowalski, G.: Information Retrieval Systems – Theory and Implementation. Kluwer Academic Publishers, Boston/Dordrecht/London (1997)

Efficient Randomized Algorithms for Text Summarization

Ahmed Mohamed and Sanguthevar Rajasekaran

Department of Computer Science & Engineering,
University of Connecticut, Storrs, CT 06268
{amohamed, rajasek}@engr.uconn.edu

Abstract. Text summarization is an important problem since it has numerous applications. This problem has been extensively studied and many approaches have been pro-posed in the literature for its solution. One such interesting approach is that of posing summarization as an optimization problem and using genetic algorithms to solve this optimization problem. In this paper we present elegant randomized algorithms for summarization based on sampling. Our experimental results show that our algorithms yield better accuracy than genetic algorithms while significantly saving on time. We have employed data from Document Understanding Conference 2002 and 2004 (DUC-2002, DUC-2004) in our experiments.

1 Introduction

Document summarization has been the focus of many researchers for the last decade, due to the increase in on-line information and the need to find the most important information in a (set of) document(s). There are different approaches to generate summaries depending on the task the summarization is required for. Summarization approaches usually fall into 3 categories (Mani and Maybury, 1999):

- *Surface-level* approaches tend to represent information in terms of shallow features, which are then selectively combined together to yield a salience function used to extract information;
- *Entity-level* approaches build an internal representation for text, modeling text entities and their relationships. These approaches tend to represent patterns of connectivity in the text (e.g., graph topology to help determine what is salient);
- *Discourse-level* approaches model the global structure of the text, and its relation to communicative goals.

Some approaches mix between two or more of the features of the above mentioned approaches, and the approaches discussed in this paper fall in that category, since they involve both surface and entity levels' features.

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 195-200

Acknowledgments This project was funded by Bell University Laboratories (BUL) and the Canada Natural Science and Engineering Research Council (NSERC). The authors would also like to thank to anonymous referees for their comments.

References

1. Green, B.F., Wolf, A.K., Chomsky, C., Laughery, K.: Baseball: An Automatic Question Answerer. In: Proceedings of the Western Joint Computer Conference. (1961) 219–224
2. Woods, W.: Progress in Natural Language Understanding: An Application to Lunar Geology. In: AFIPS Conference Proceeding. Volume 42. (1973) 441–450
3. Mollá, D., Vicedo, J.L., eds.: Workshop on Question Answering in Restricted Domains - ACL-2004, Barcelona, Spain (2004)
4. Mollá, D., Vicedo, J.L., eds.: AAAI-05 Workshop on Question Answering in Restricted Domains, Pittsburg, Pennsylvania (2005) to appear.
5. Mollá, D., Vicedo, J.L., eds.: Special Issue of Computational Linguistics on Question Answering in Restricted Domains. (2005) to appear.
6. Herzog, O., Rollinger, C.R., eds.: Text Understanding in LILOG, Integrating Computational Linguistics and Artificial Intelligence, Final Report on the IBM Germany LILOG-Project. In Herzog, O., Rollinger, C.R., eds.: Text Understanding in LILOG. Volume 546 of Lecture Notes in Computer Science., Springer (1991)
7. Mollá, D., Berri, J., Hess, M.: A real world implementation of answer extraction. In: Proceedings of the 9th International Workshop on Database and Expert Systems, Workshop: Natural Language and Information Systems (NLIS-98), Vienna (1998)
8. Benamara, F., Saint-Dizier, P.: Advanced Relaxation for Cooperative Question Answering. In: New Directions in Question Answering. (2004) 263–274
9. Oroumchian, F., Darrudi, E., Ofoghi, B.: Knowledge-Based Question Answering with Human Plausible Reasoning. In: Proceedings of the 5th International Conference on Recent Advances in Soft Computing. (2004)
10. Light, M., Mann, G., Riloff, E., Breck, E.: Analyses for Elucidating Current Question Answering Technology. Natural Language Engineering **7** (2001)
11. Diekema, A.R., Yilmazel, O., Liddy, E.D.: Evaluation of Restricted Domain Question-Answering Systems. In: Proceedings of the Association of Computational Linguistics 2004 Workshop on Question Answering in Restricted Domains (ACL-2004), Barcelona, Spain (2004)
12. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M.: OKAPI at TREC-3. In D.K.Hartman, ed.: Overview of the Third TExt Retrieval Conference (TREC-3), Gaithersburg, NIST (1995) 109–130
13. Doan-Nguyen, H., Kosseim, L.: Improving the Precision of a Closed-Domain Question-Answering System with Semantic Information, Avignon, France, RIAO-2004 (2004)
14. Doan-Nguyen, H., Kosseim, L.: The Problem of Precision in Restricted-Domain Question-Answering. Some Proposed Methods of Improvement, Barcelona, Spain, ACL (2004)
15. Kowalski, G.: Information Retrieval Systems – Theory and Implementation. Kluwer Academic Publishers, Boston/Dordrecht/London (1997)

Efficient Randomized Algorithms for Text Summarization

Ahmed Mohamed and Sanguthevar Rajasekaran

Department of Computer Science & Engineering,
University of Connecticut, Storrs, CT 06268
{amohamed, rajasek}@engr.uconn.edu

Abstract. Text summarization is an important problem since it has numerous applications. This problem has been extensively studied and many approaches have been pro-posed in the literature for its solution. One such interesting approach is that of posing summarization as an optimization problem and using genetic algorithms to solve this optimization problem. In this paper we present elegant randomized algorithms for summarization based on sampling. Our experimental results show that our algorithms yield better accuracy than genetic algorithms while significantly saving on time. We have employed data from Document Understanding Conference 2002 and 2004 (DUC-2002, DUC-2004) in our experiments.

1 Introduction

Document summarization has been the focus of many researchers for the last decade, due to the increase in on-line information and the need to find the most important information in a (set of) document(s). There are different approaches to generate summaries depending on the task the summarization is required for. Summarization approaches usually fall into 3 categories (Mani and Maybury, 1999):

- *Surface-level* approaches tend to represent information in terms of shallow features, which are then selectively combined together to yield a salience function used to extract information;
- *Entity-level* approaches build an internal representation for text, modeling text entities and their relationships. These approaches tend to represent patterns of connectivity in the text (e.g., graph topology to help determine what is salient);
- *Discourse-level* approaches model the global structure of the text, and its relation to communicative goals.

Some approaches mix between two or more of the features of the above mentioned approaches, and the approaches discussed in this paper fall in that category, since they involve both surface and entity levels' features.

© *A. Gelbukh (Editor)*
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 195-200

2 Background and Related Work

2.1 ExtraNews

In our study, we considered the work done at LARIS laboratory (Fatma et al., 2004). In this approach (called ExtraNews) summarization is considered as an optimization problem. A set of summaries is generated randomly and then a Genetic Algorithm is utilized to come up with a good summary. They use a fitness function that depends on three different factors. The first factor ω1 is related to the length of the summary, in which the length of the summary is tested against the required target length, as shown in the following equation:

$$\omega_1 = \begin{cases} \dfrac{\sum_{i=1}^{m} L(ph_i)}{L_E} & \text{if } \sum_{i=1}^{m} L(ph_i) < 0.9 \times L_E \\ 0 & \text{if } \sum_{i=1}^{m} L(ph_i) > L_E \end{cases} \quad (1)$$

where $L(ph_i)$ is the length of sentence i; L_E is the target summary length set by the user and m is the number of sentences in the summary.

The second factor ω_2 pertains to the coverage criterion. It calculates how many of the original keywords have been captured in the target summary:

$$\omega_2 = \frac{\sum M_{ext}}{\sum M_{doc}} \quad (2)$$

where M_{ext} represents the keywords in the summary and M_{doc} represents the keywords in the source document-set.

The last factor ω_3 is associated with the weight criterion. It is the fraction of the sum of weights of all sentences in the summary to the maximal summary weight in the population:

$$\omega_3 = \frac{\sum P_{ext}}{Max(P_{pop})} \quad (3)$$

where P_{ext} is the weight of a sentence of the summary and $Max(P_{pop})$ is the maximum summary weight in the population. It was not mentioned, however, in (Fatma et al., 2004) how the weight of the sentence is calculated. For this reason, we choose to use a cosine similarity measure (Salton et al., 1997) to weight each sentence, in which the summary and each sentence in the summary are represented as vectors of terms and then the weight is calculated from the following formula:

$$sim(D1, D2) = d1 \bullet d2 \tag{4}$$

where: D_1, D_2 are documents 1 and 2 respectively, and d_1, d_2 are the term vectors of documents 1 and 2 respectively..

It is also important to note that (Fatma et al., 2004) did not mention how the above three coefficients were composed together to form the fitness function. So, we assumed that the fitness function is the product of all coefficients.

ExtraNews system ranked very well in tasks 4 (creating a short summary for English translations of a document cluster) and 5 (creating a short summary from a document cluster answering the question "Who is X?" where X is a name of a person/Group of people) of the Data Understanding Conference (DUC-2004) tasks. It ranked above average in task 2 (creating a short summary for each document cluster) and in task 3a (creating a very short summary for automatic English translation of a document cluster). However, it ranked badly in tasks 1 (creating a very short summary for English translation of a document cluster) and 3b (creating a very short summary for manual English translation of a document cluster) (Over, 2002). They attributed the last bad results to the fact that the phrases considered in the segmentation process are not very well suitable for very short summaries.

2.2 Our Randomized Algorithms

Randomized algorithms have played a vital role in the past three decades in solving many fundamental problems of computing efficiently. Many problems have been shown to be better solvable using randomization than determinism. For examples see (Horowitz, Sahni and Rajasekaran 1998). Algorithms such as simulated annealing that have proven very effective in solving some intractable problems in practice are examples of randomized algorithms. Both in practice and theory randomization has resulted in the design of efficient algorithms.

One popular theme in randomization has been that of sampling. In its simplest form sampling can be defined as follows. Say we want to measure a certain characteristic C from a dataset D. We could do this by processing all the points in D. Alternatively we could pick a random subset D' (called the sample) of D, measure the same characteristic in D', and from this sample measurement infer the value of the characteristic in D. Preferably, we should be able to infer this value with high probability. For a survey of sampling techniques see (Rajasekaran and Krizanc 2001). Our algorithm for summarization is based on sampling. Before presenting our algorithm, we briefly describe the approach taken in the ExtraNews system. This sys-tem employs genetic algorithms.

The genetic algorithm for solving any optimization problem has been motivated by Darwin's theory of evolution and works as follows. A population of random points from the feasible space is chosen at the beginning. The 'fitness' of each point is computed. A new population is obtained from the old one using two operators, namely, crossover and mutation. A crossover operation refers to taking two points in the population and producing an 'offspring' point similar to the way an offspring chromosome is produced from two parent chromosomes. Crossovers are per-formed typically be-

tween pairs with high fitness values (with the hope that the offspring will be fitter). The above process of producing a new population from an old one is repeated until certain conditions are satisfied. For example, the algorithm could terminate after producing a certain number of populations or when the best solution in the population does not change significantly (from one population to the next).

Fatma *et al.* have employed genetic algorithms in the context of summarization as follows. They first produce a population of random points. Each point is nothing but a summary formed by random sentences picked. Fitness of each point is calculated. A new population is then produced by using the GA operators (crossover and mutation) from the older population and the newer population replaces the older one, and so on. Every time a new population is formed, the best summary is kept in a safe place until a better summary is found, then the better summary will replace the poorer summary.

We propose a randomized algorithm based on sampling. We pick a random sample as in (Fatma *et al.*) and choose the best summary in this sample. The process stops after two generations only. We employ the three criteria that Fatma *et al.* have employed for measuring fitness of summaries.

It is important to mention that our approach uses the same fitness function used with the GA as a built-in function. So, the GA is not required to run in conjunction with our approach.

3 Data and Experimental Design

3.1 Data

We used multi-document extracts from DUC-2002 and from DUC-2004 (task 2) in our experiment. In the corpus of DUC-2002, each of the ten information analysts from the National Institute of Standards and Technology (NIST) chose one set of newswire/paper articles in the following topics (Over 2002):

- A single natural disaster event with documents created within at most a 7-day window;
- A single event of any type with documents created within at most a 7-day window;
- Multiple distinct events of the same type (no time limit);
- Biographical (discuss a single person);

Each assessor chose 2 more sets of articles so that we ended up with a total of 15 document sets of each type. Each set contains about 10 documents. All documents in a set are mainly about a specific "concept."

The corpus of DUC-2004 (task 2) is composed of 50 TDT English news clusters. Each cluster contains about 10 documents chosen by NIST about one single event (Over and Yen, 2004).

3.2 Experimental Design

A total of 59 document-sets from DUC-2002 and 50 document-sets from DUC-2004 have been used in our experiment to investigate the performance of our randomized algorithm. We ran the Genetic Algorithm (GA) on both corpuses. Since we are comparing our algorithm's performance to that of the GA, we found it more meaningful to use the fitness function (Fatma et al., 2004) used in their GA to evaluate the quality of our summaries as well.

4 Results

Tables 1 and 2 show a comparison between the results obtained using DUC-2002 and DUC-2004, respectively, of GA and our RA performance in terms of quality, number of times summaries produced faster, average quality and average time spent by each algorithm, respectively. The experimental results show that the randomized algorithm produced competitive results in much less time than the Genetic Algorithm.

Table 1. Comparison between GA and RA results from the DUC-2002 data.

	GA	RA
# of best summaries	5/59	54/59
# of summaries produced faster	0/59	59/59
Average quality	0.129	0.152
Average time (sec)	23.1	13.6

Table 2. Comparison between GA and RA results from the DUC-2004 data.

	GA	RA
# of best summaries	8/50	42/50
# of summaries produced faster	0/50	50/50
Average quality	0.041	0.051
Average time (sec)	8.4	3.5

5 Conclusions

In this paper we have presented an elegant randomized algorithm for text summarization. This algorithm is based on sampling. Our algorithm has been compared with the Genetic Algorithm of (Fatma et al. 2004). This comparison shows that our randomized algorithm produces summaries that are comparable in quality to those produced by GA while taking much less time. An important open problem is to study if sampling can be used in conjunction with other text summarization approaches to obtain similar

speedups. We are also planning on testing our approach against the GA approach on the DUC-2004 collection when it is ready for experimentation.

Acknowledgment. SR has been supported in part by the NSF Grant ITR: 0326155.

References

1. J. K. Fatma, J. Maher, B. H. Lamia, B. H. Abdelmajid, LARIS Laboratory. 2004. Summarization at LARIS Laboratory. Document Understanding Workshop. May 6-7, 2004, Boston Park Plaza Hotel and Towers, Boston, USA
2. E. Horowitz, S. Sahni and S. Rajasekaran, Computer Algorithms, W. H. Freeman Press, 1998.
3. Chin-Yew Lin and Eduard Hovy. 2002. Manual and Automatic Evaluation of Summaries. In Proceedings of the Workshop on Automatic Summarization post conference workshop of ACL-02, Philadelphia, PA, U.S.A., July 11-12 (DUC2002).
4. Inderjeet Mani and Mark T. Maybury. (1999). Advances in Automatic Text Summarization. The MIT Press.
5. Paul Over and James Yen. 2004. An Introduction to DUC 2004 Intrinsic Evaluation of Generic New Text Summarization Systems. Document Understainding Conferences website (http://www-nlpir.nist.gov/projects/duc/)
6. Paul Over and Walter Liggett. 2002. Introduction to DUC-2002: an Intrinsic Evaluation of Generic News Text Summarization Systems. Document Understand-ing Conferences website (http://duc.nist.gov/)
7. Sanguthevar Rajasekaran and Danny Krizanc, Random Sampling: Sorting and Selection, in Handbook of Randomized Computing, Kluwer Academic Press, 2001.
8. Gerard Salton. 1988. Automatic Text Processing. Addison-Wesley Publishing Company.
9. Gerard Salton, Amit Singhal, Mandar Mitra, and Chris Buckley. 1997. Automatic Text Structuring and Summarization. Information Processing and Management 33(2): 193-207. Reprinted in Advances in Automatic Text Summarization, I. Mani and M.T. Maybury (eds.), 341–35. Cambridge, MA: MIT Press.

Computer-Assisted Language Learning

Arabic Error Feedback in an Online Arabic Learning System

Khaled F. Shaalan[1,2] and Habib E. Talhami[1,2]

[1] Institute of Informatics
The British University in Dubai
P. O. Box 502216, Dubai,UAE
[2] Honorary Fellow, School of Informatics, University of Edinburgh
{khaled.shaalan,habib.talham}@buid.ac.ae

Abstract. Arabic is a Semitic language that is rich in its morphology and syntax. The very numerous and complex grammar rules of the language could be confusing even for Arabic native speakers. Many Arabic intelligent computer-assisted language-learning (ICALL) systems have neither deep error analysis nor sophisticated error handling. In this paper, we report an attempt at developing an error analyzer and error handler for Arabic as an important part of the Arabic ICALL system. In this system the learners are encouraged to construct sentences freely in various contexts and are guided to recognize by themselves the errors or inappropriate usage of their language constructs. We used natural language processing (NLP) tools such as a morphological analyzer and a syntax analyzer for error analysis and to give feedback to the learner. Furthermore, we propose a mechanism of correction by the learner, which allows the learner to correct the typed sentence independently. This will result in the learner being able to figure out what the error is. Examples of error analysis and error handling will be given and will illustrate how the system works.

1 Introduction

Computer-assisted language learning (CALL) addresses the use of computers for language teaching and learning. CALL emerged in the early days of computers. Since the early 1960's, CALL systems have been designed and built. The effectiveness of CALL systems has been demonstrated by many researchers [6, 7]. More than a decade ago, Intelligent Computer-Assisted Language Learning (ICALL) started as a separate research field, when artificial intelligence (AI) technologies were mature enough to be included in language learning systems. The beginning of the new research field was characterized by intelligent tutoring systems (ITS), which embedded some NLP features to extend the functionality of traditional language learning systems. The continuous advances in ICALL systems have been documented in several publications [2, 3, 5, 9].

On of the weaknesses of current Arabic ICALL systems is that learners cannot key in Arabic sentences freely. Similarly, the system cannot guide the learner to correct the most likely ill-formed input sentences. The learner just accepts the information,

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 203-212

which has been pre-programmed into the system. For these systems to be useful, more research to combine NLP techniques with language learning systems is needed [8]. Parsing, the core component in ICALL systems, allows the system both to analyze the learner's input and to generate responses to that input [4]. Allowing learners to phrase their own sentences freely without following any pre-fixed rules can improve the effectiveness of ICALL systems, especially when the expected answers are relatively short and well-focused [1]. Both the well- and ill-formed structure of the input sentence can be recognized. The learner should be allowed to correct the typed sentence independently.

This paper describes error analysis and handling in an Arabic ICALL system using NLP techniques, which is a step towards enhancing current Arabic ICALL systems. The current system guides learners to recognize by themselves the errors or improper usage of their language constructs. In other words, it helps learners to learn from their own mistakes. It doesn't give them the correct answer directly but it enables them to try over and over again. In this system, we use NLP tools such as a morphological analyzer, a syntax analyzer, and an error analyzer to give feedback to the learner. Furthermore, we propose a mechanism of correction by learners which allows the learner to correct the typed sentence independently.

The rest of the paper is organized as follows: Arabic ICALL framework is summarized in Section 2. This is followed by a description of the Arabic sentence analysis in Section 3. Next, the proposed feedback component responsible for error analysis and error handling is described in Section 4. Finally, a conclusion and recommendations for further enhancements are given in Section 5.

2 The Arabic ICALL Framework

Figure 1 shows the overall framework of the proposed Arabic ICALL system by [8]. This system consists of the following components: user interface, course material, sentence analysis, and feedback. The user interface provides the means of communications between the learner and the Arabic ICALL system. The course material includes educational units, an item (question) bank, a test generator, and an acquisition tool. The sentence analysis includes a morphological analyzer, a syntax analyzer (parser), grammar rules, and a lexicon. The feedback component includes an error analyzer and error handler that are used to parse ill-formed learner input and to issue feedback to the learner.

3 Arabic Sentence Analysis

The sentence analysis in Arabic ICALL includes a morphological analyzer, a syntax analyzer (parser), grammar rules, and a lexicon. The sentence analysis works as follows. The learner written input is first fed into the interactive preprocessor, where it is grouped into words. The words in the input are then decomposed into roots and affixes by the morphological analyzer, which obtains information about the subparts from the lexicon (e.g., part of speech, number, case). The subparts so identified are

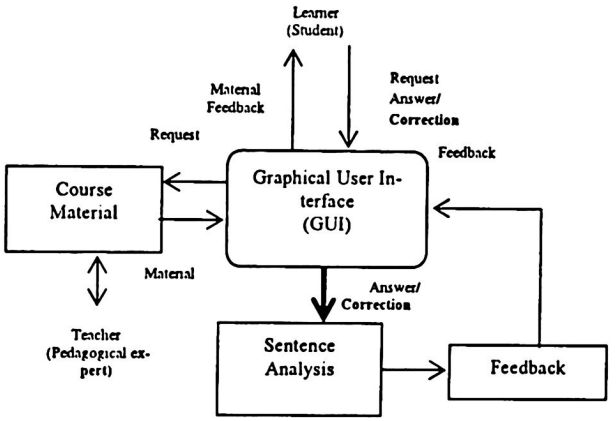


Figure 1. Overall Framework of the Proposed Arabic ICALL System

then reunified into whole words and passed along to the syntactic parser. The Arabic parser, which is based on Definite Clause Grammar (DCG) formalism, tries to build a structure (usually, 'parse tree') based on the information from the lexicon concerning the grammatical relations between the words. The parser then applies a set of rules representing the grammar of Arabic until it finds the structure represented by the input sentence. This structure is passed to the feedback component that is equipped with an error analyzer and an error handler. The error analyzer identifies and records any errors made in the structural description which is generated. The error handler applies the error handling mechanism on the ill-formed leaner input to produce the appropriate feedback message.

3.1 The Grammar Formalism

Arabic grammar in Arabic ICALL is written in the DCG formalism. There two types of grammars that have been used for learning Arabic: Grammar rules for grammatically correct sentence and grammar rules for linguistic analysis (Iarab).

In the following, we show an excerpt of DCG rules used for parsing a grammatically correct Arabic verbal sentence.

```
(1)  verbal_sentence --> simple_verbal_sentence.
(2)  verbal_sentence --> prefixed_verbal_sentence.
(3)  verbal_sentence --> special_verbal_sentence.
(4)  simple_verbal_sentence --> verb, subject, object,
     unrestricted_object.
```

For simplicity, these rules do not include linguistic features such as gender, number and definition, which are assigned to each non-terminal. Rule (4) illustrates a grammar rule for parsing a simple verbal sentence that consists of four constituents: a verb, a subject, an object and an unrestricted object "قلطم لوعفم". An unrestricted object is a noun that originates from the infinitive verb. This kind of repetition is considered a

mark of good style. In Arabic, repeating the verbal noun after the verb makes the sentence more emphatic. This is explained by the following example:

ةمیظع ةدعاسم ىندعاس

He helped me a great deal of help

In the following, we show an excerpt of DCG rules used for the linguistic analysis of the words between brackets. These words are an object that is followed by an adjective such that the adjective agrees with the object (the noun it modifies) in number, gender, definition, and end case.

```
object(Words_bet_brackets, Rest, Analysis) -->
 [Object], [Adjective],
 {get_analysis(Object, Gender, Num, Def,
                Words_bet_brackets, Rest1, End_case,
                Analysis1, 'هب لوعفم'),
   get_analysis(Adjective, Gender, Num, Def, Rest1,
                 Rest, End_case, Analysis2, 'نعت'),
   append(Analysis1, Analysis2, Analysis)
 }.
```

As an example, this rule can be used with the following question:

أعرب ام بين القوسين في الجملة الآتية:
أفل العقاد (ابتك ثيرة).

Give the linguistic analysis of the words between brackets in the following sentence: Al-aakad authored (many books).

The parser produces a parse representing the linguistic analysis of the Arabic word that consists of a quadruple abstract representation of the canonical form:

السبب	+	علامة الإعراب	+	الإعراب	+	الموقع الإعرابي
Reason	+	Analytic sign	+	End case	+	Analytic location

4 The Feedback Component

Feedback is the computer's response to answers made by learners. We have augmented the Arabic grammar with heuristic rules (buggy rules) which are capable of parsing ill-formed input and which apply if the grammatical rules fail. The feedback system compares the analysis of the learner's answer with the correct answer that is generated by the system. If there is a match, a positive message will be sent to the learner. Otherwise, a feedback message will be sent to the learner. In the following subsections, we show how the system catches the learner's errors and how it handles the ill-formed natural language input.

4.1 Rules for Error Analysis

In our implementation, we have augmented the Arabic grammar with heuristic rules which are capable of parsing ill-formed input (buggy rules) and which apply if the

grammatical rules fail. As an example, consider the following question to complete a sentence with a suitable unrestricted object "المفعول المطلق":

أكمل بمفعول مطلق مناسب:
أبر أبى _____ .

Complete the following with the correct unrestricted object:
I am kind to my father _____.

The following is an analysis of the possible learner's answer along with the corresponding feedback:

- A word that is not a noun. Issue a message describing that the unrestricted object should be a noun.
- A word that is a noun but does not originate from the infinitive verb. Issue a message describing that the unrestricted object should be the infinitive of the verb.
- A word that is both a noun and originates from the infinitive verb but is defined. Issue a message describing that the unrestricted object should be undefined.
- A word that is a noun, originates from the infinitive verb but needs the end case "Alef Tanween", and is undefined. Issue a message describing that a missing end case of the unrestricted object.
- A Correct answer. Issue a positive message.

4.2 Error Handling Mechanism

Learner's responses which have special handling mechanisms in case of ill-formed learner input are: linguistic analysis, classification into categories, sentence transformation, and completing a sentence.

Handling of linguistic analysis. Linguistic analysis questions can apply either for an entire sentence or a part of it. The latter is usually a sequence of words between brackets. The following description outlines the steps for handling of linguistic analysis:

- Parse the given sentence (or the sequence of words between brackets) and generate its linguistic analysis in a quadruple abstract representation form
- Convert learner answer into the abstract representation form
- Compare the learner's answer with the generated answer to issue the appropriate feedback message

Example:

هل ترى فرقا إعرابيا فيما بين القوسين؟
- استمتعت بجو الريف (استمتاعا).
- أذهب إلى الريف (استمتاعا) بجوه.

What's the difference in linguistic analysis of the words in parentheses?
- *I enjoyed the country weather (very much)*
- *I go to the countryside (to enjoy) its weather*

The parser is used to analyze each of the input sentences. The generated correct linguistic analyses of the words in parentheses are the following:

- First word: ['قلطم لوعفم', 'ابوصنم', 'فتحة', 'درفم'] [unrestricted object, accusative, fat-hah, singular]
- Second word:['فلجأ لوعفم', 'ابوصنم', 'فتحة', 'درفم'] [causative object, accusative, fat-hah, singular]

The difference, in this case, is in the analytic location (i.e. the first argument in the quadruple abstract representation). The learner's answer is also converted to the quadruple abstract representation. The comparison between these representations will issue the appropriate feedback that describes the source of the error. The possible source of the errors could be: incorrect analytic location, incorrect end case, incorrect reason, or a partially correct answer.

Handling of classification into categories. Classification into categories questions can apply either for identifying morphological categories or for identifying syntactic constituents (possibly, a complete sentence). The following description outlines the steps for handling classification into morphological categories:

- Morphologically analyze the words in the given sentence and determine the words features.
- Generate N lists, a classification of the words according to the questions words.
- Assign the learner answer to N Lists.
- Compare the learner's answer with the generated answer to issue the appropriate feedback message

Example:

عين الاسم او الفعل او الحرف في الجملة الآتية:
وقف التلاميذ احتراما للمعلم

Identify the category of each of the words in following sentence
The students stood up respecting the teacher

The morphological analyzer is used to analyze each inflected Arabic word to recognize its category. Then, according to the word category, the words are classified into three lists. The following is the generated correct answer:

- Verb: [['وقف',verb,male,singular,past,...]] (stood)
- Noun:[['المعلم',noun,male,singular,def,...], ['احتراما',noun, male,singular,undef, ...],
- ['التلاميذ',noun, male,plural,def, ...]] (the teacher, honoring, the students)
- Particle: [['ال',particle,def_article,...]] (the)

The learner's answer is also assigned to three lists containing verbs, nouns, and particles, respectively. The comparison between the corresponding lists will issue the appropriate feedback that describes the source of the error. The possible source of the errors could be: missing words from the respective morphological category, or assigning a word to an incorrect morphological category.

The following description outlines the steps for handling of classification into syntactic constituents:

- Parse the given sentence and determine the parse tree.
- Generate N lists, a classification of the sentence's constituents according to the questions words
- Assign the learner answer to N Lists.

Compare the learner's answer with the generated answer to issue the appropriate feedback message.

Example:

عين المبتدأ والخبر بمبتدأ نوعه في الجملة الآتية:
الجنود الشجعان يتنصرون

Identify the inchoative and enunciative, and the type of the enunciative in the following sentence.
the brave soldiers fought to victory

The parser is used to analyze the input sentence into a parse tree as follows:

```
nominal_sentence(
 inchoative(noun('الجنود',noun, male,plural,def,...),
                    adj('الشجعان', noun,
 male,plural,def,adj,...)),
 enunciative(verbal_sentence(verb('ينتصرون',verb, male,
 plural,present,...)))
 )
```

Then, according to the parse tree, the words are classified into three lists. The following is the generated correct answer:

- inchoative: [noun('الجنود', noun, male,plural,def,...),adj('الشجعان', noun, male,plural,def,adj,...)] (the brave soldiers)
- enunciative: [verbal_sentence(verb('ينتصرون', verb,male, plural,present,...))] (make victory)
- enunciative type: [verbal_sentence]

The learner's answer is also assigned to three lists containing inchoative, enunciative, and enunciative type, respectively. The comparison between the corresponding lists will issue the appropriate feedback that describes the source of the error. The possible source of the errors could be: incorrect constituent type (analytic location), or assigning a syntactic constituent to an incorrect category.

Handling of transformation of a sentence. Transformation questions require the learner to change/rewrite the form of a sentence. The following description outlines the steps for handling of transformation of a sentence:

- Parse the given sentence and determine the parse tree; apply a tree-to-tree transformation to generate the transformed parse tree
- Parse the learner's answer to determine the parse tree.
- Compare the parse tree of the learner's answer with the parse tree of the generated answer to issue the appropriate feedback message.

Example:

الجملة الآتية اسمية أجعلها فعلية:
المعلم يشرح الدرس

Change the following nominal sentence into verbal sentence:
The teacher teaches the lesson

The parser is used to analyze the input sentence into a parse tree as follows:

```
nominal_sentence(
  inchoative(noun('المعلم',noun,male,singular,def,...)),
  enunciative(verbal_sentence(verb('حرشي',verb,male,
   singular,present,...),object(noun('الدرس',noun,
   male,singular,def,...))))
  )
```

Then, the parse tree of the nominal sentence is transformed to the following verbal sentence.

```
verbal_sentence(
  verb('حرشي',verb,male,singular,present,...),
  subject(noun('المعلم',noun,male,singular,def,...)),
  object(noun('الدرس', noun,male,singular,def,...))
  )
```

In addition, words of the transformed parse tree is grouped in a list as follows:

List of words: [verb('حرشي',verb,male,singular,present,...), noun('المعلم', noun,male,singular,def,...),noun('الدرس', noun,male,singular,def,...)]

The learner's answer is also analyzed into a parse tree and words are grouped into a list. The comparison between the corresponding representations will issue the appropriate feedback that describes the source of the error. The possible source of errors could be: extra words, missing words, grammatically incorrect sentence, or incorrect transformation of a word (incorrect verb: tense, number, gender,...; incorrect noun: number, gender, definition,...).

Handling of Fill-in-the-blanks. Fill-in-the-blank questions can apply for the generation of isolated words with a particular form, or to the generation of words to complete a sentence that achieves feature agreement among its constituents. The following description outlines the steps for handling of rewriting of isolated words with particular morphological form:

- Morphologically generate the given words and determine their features.
- Morphologically analyze the learner's answer and determine the words features.
- Compare the parse tree of the learner's answer with the parse tree of the generated answer to issue the appropriate feedback message.

Example:

س واجمع اثنّ املا الكلمات الآتية:
- المهندس
- الثمرة
- صحراء

What is the correct dual and regular plural of the following words?
- *Engineer*
- *Fruit*
- *Desert*

The morphological generator is used to synthesize the given words into the dual and plural forms taking into consideration the possible end case. The following is the generated correct answer:

- Dual list: [['الممتلئان',noun,male,dual,def,nominal,...], [['الممتلئين',noun,male, dual,def,accusative, ...], ['الممتلئتان', noun,female,dual,def,nominal, ...], ['الممتلئتين', noun,female,dual,def,accusative, ...], ['فارغتان', noun, female,dual, undef,nominal, ...],['فارغتين', noun, female,dual,undef, accusative, ...]]
- Plural list: [['الممتلئون',noun,male,plural,def,nominal, ...], [['الممتلئين', noun,male,plural,def,accusative, ...], ['الممتلئات',noun,female,plural,def, ...], ['فارغات', noun, female, plural,undef,...]]

The morphological analyzer is used to analyze each inflected Arabic word of the learner's answer. These words are classified into two lists containing dual and plural forms, respectively. The comparison between the corresponding lists will issue the appropriate feedback that describes the source of the error. The possible source of errors could be: different word, incorrect word category (switch dual forms with plural forms), incorrect generation of a word (incorrect verb: tense, number, gender,...; incorrect noun: number, gender, definition,...).

The following description outlines the steps for handling of fill-in-the-blank to achieve agreement among sentence constituents:

- Analyze the sentence and determine its constituents.
- Morphologically generate the missed words.
- Parse the learner's answer to determine the parse tree.
- Compare the parse tree of the learner's answer with the parse tree of the generated answer to issue the appropriate feedback message.

Example:

أكمل بما هو مطلوب للجملة:
أنا أحب أبي _____.

Complete the sentence with an unrestricted noun that makes the sentence more emphatic:
I am kind with my father _____.

The parser is used to analyze the partial input sentence and determine its constituents as follows:

```
verbal_sentence(
 verb('أحب',verb,male,singular,present,intrans,'حب'...),
       subject(noun('أبي', noun,male,singular,undef,...))
 )
```

The morphological generator uses the infinitive verb 'حب' (kindness) of the main verb to synthesize the unrestricted object 'احب' (extremely kind). The learner's answer is also analyzed into a parse tree. The comparison between the corresponding representations will issue the appropriate feedback that describes the source of the error. The possible source of errors could be: different word (sense or category), incorrect morphological generation of a word (incorrect verb: tense, number, gender, ...; incorrect noun: number, gender, definition,...), incorrect syntactic generation of a word (s not in emphatic form, does not originates from the infinitive verb, ...).

5 Conclusions and Future Work

In this paper, we have discussed issues related to the development of error analysis and error handling in Arabic ICALL systems. NLP tools can be useful for ICALL and in particular for giving meaningful feedback to the learner. Learner-system communication in free natural language is computationally the most challenging and pedagogically the most valuable scenario in Arabic ICALL. The deep syntactic analysis of the learner's answer, whether correct or wrong, is compared against a system-generated answer. This enables feedback elaboration that helps learners to understand better and fill in the knowledge gaps.

The rule-based approach is used to give some freedom to the language learners in the way they phrase their answers. This also enables the exercise author to enter only one possible correct answer, thus saving much time compared to the previous pattern matching answer coding approach. The present system has been implemented using SICStus Prolog. The system is transportable and capable of running on an IBM PC which allows the learner to use it to learn Arabic language anywhere and anytime.

We plan to enrich the present system, e.g. make the system available on the Internet to serve remote learners worldwide (especially learners of Arabic as a second language), and extend the grammar coverage to include more advanced grammar levels.

References

1. Boytcheva, S.,Vitanova, I., Strupchanska, A., Yankova, M., Angelova, G.: Towards the assessment of free learner's utterances in CALL. In the Proceedings of InSTIL/ICALL2004 – NLP and Speech Technologies in Advanced Language Learning Systems, Venice, 2004.
2. Cameron, K. (editor): call-media, design, & applications, Swets & Zeitlinge, 1999.
3. Gamper, J.,Knapp, J.: A Review of Intelligent CALL Systems. Computer Assisted Language Learning (CALL): An International Journal, 15(4), 329-342, Belgium: SWETS & ZEITLINGER publisher, 2002.
4. Holland, M. V., Maisano, R. , Alderks, C.: Parsers In Tutors: What Are They Good For? CALICO Journal, 11(1), 28-46, 1993.
5. Holland, M. V., Kaplan, J.D.: Sama, M.R., editors, intelligent language tutors: theory shaping technology. mahwah, New Jersey: Lawrence Erlbaum Associates, Inc., 1995.
6. Lam, F.S., Pennington, M.C.: The Computer vs. the Pen: A Comparative Study of Word Processing in a Hong Kong Secondary Classroom. Computer Assisted Language Learning (CALL): An International Journal, 8(1), 1995, 75-92, Belgium: SWETS & ZEITLINGER publisher.
7. McEnery, T., Baker, J.P., Wilson, A.: A Statistical Analysis of Corpus Based Computer vs. Traditional Human Teaching Methods of Part of Speech Analysis. Computer Assisted Language Learning (CALL): An International Journal, 8(2), 1995, 259-274, Belgium: SWETS & ZEITLINGER publisher.
8. Shaalan, K.: An Intelligent Computer Assisted Language Learning System for Arabic Learners, Computer Assisted Language Learning: An International Journal, 18(1 & 2), PP. 81-108, Taylor & Francis Group Ltd., February 2005.
9. Swartz, M., Yazdani, M. (editors): intelligent tutoring systems for foreign language learning. chapter Introduction. Springer-Verlag, 1992.

Author Index
Índice de autores

Abe, Akinori	125
Aguilar, A.	83
Alexandrov, Mikhail	27
Amores, Gabriel	3
Blanco, Xavier	27
Deng, Xiaotie	151
Ferrández, Antonio	83, 177
Ferrández, Sergio	83, 177
Fliedl, Günther	49
Gelbukh, Alexander	27
González, David	3
Hai, Doan-Nguyen	183
Han, Song	151
Hara, Kazuma	105
Huang, Joshua	139
Isahara, Hitoshi	163
Johansson, Christer	57
Johnsen, Lars G.	57
Jung, Hanmin	15
Kanamaru, Toshiyuki	163
Kita, Kenji	115
Kogure, Kiyoshi	125
Koo, Hee-Kwan	15
Kop, Christian	49
Kosseim, Leila	183
Kuroiwa, Shingo	105, 115
Kuwahara, Noriaki	125
Lalitha Devi, Sobha	37
Legrand, Steve	71
Li, Yan	139
López-Moreno, P.	83
Manchón , Pilar	3
Mayr, Heinrich C.	49
Mohamed, Ahmed	195
Murata, Masaki	163
Ozaku, Hiromi itoh	125
Palanisamy , Arulmozhi	37
Park, Dong-In	15
Pérez, Gillermo	3
Rajasekaran, Sanguthevar	195
Ren, Fuji	105, 115
Roger, S.	83
Saba, Walid S.	93
Sagara, Kaoru	125
Shaalan, Khaled F.	203
Shishibori, Masami	115
Sung , Won-Kyung	15
Talhami, Habib E.	203
Tanaka, Koji	105
Tsuge, Satoru	105, 115
Utiyama, Masao	163
Vöhringer, Jürgen	49
Wang, Fu Lee	151
Weber , Georg	49
Winkler, Christian	49
Xu, Xiaofei	139
Ye, Yunming	139
Zou, Feng	151

Editorial Board of the Volume

Comité editorial de volumen

Eneko Agirre
Christian Boitet
Igor Bolshakov
Nicoletta Calzolari
John Carroll
Dan Cristea
Barbara Di Eugenio
Gregory Grefenstette
Linda van Guilder
Cătălina Hallett
Yasunari Harada
Eduard Hovy
Nancy Ide
Diana Inkpen
Frederick Jelinek
Aravind Krishna Joshi
Martin Kay
Alma Kharrat
Adam Kilgarriff
Richard Kittredge
Kevin Knight
Alexander Koller
Grzegorz Kondrak
Sandra Kuebler
Ken Litkowski
Hugo Liu
Aurelio López López
Bernardo Magnini
Daniel Marcu
Carlos Martín-Vide
Igor Mel'čuk
Rada Mihalcea
Ruslan Mitkov
Masaki Murata
Vivi Nastase
Olga Nevzorova
Nicolas Nicolov
Sergei Nirenburg
Constantin Orasan
Manuel Palomar
Ted Pedersen
Viktor Pekar
Stelios Piperidis
James Pustejovsky
Fuji Ren
Fabio Rinaldi
Horacio Rodriguez
Vasile Rus
Ivan Sag
Franco Salvetti
Serge Sharoff
Grigori Sidorov
Thamar Solorio
Carlo Strapparava
Maosong Sun
John Tait
Benjamin Ka-yin T'sou
Felisa Verdejo
Karin Verspoor
Manuel Vilares Ferro
Yorick Wilks

Additional Reviewers

Árbitros adicionales

Mohamed Abdel Fattah
Mustafa Abusalah
Farooq Ahmad
Iñaki Alegria
Muath Alzghool
Bogdan Babych
Verginica Barbu Mititelu
Fco. Mario Barcala Rodríguez
Francesca Bertagna
Dimitar Blagoev

Hiram Calvo Castro
Anna Clark
Daoud Clarke
Andras Csomai
Victor Manuel Darriba Bilbao
Jeremy Ellman
Davide Fossati
Oana Frunza
Irbis Gallegos
Jorge Graña
Samer Hassan
David Hope
Scott K. Imig
Aminul Islam
Shih-Wen Ke
Stephan Kepser
Rob Koeling
Alberto Lavelli
Fennie Liang
Christian Loza
Xin Lu
Fernando Magán Muñoz
Raquel Martínez
Jaime Mendez
Dragos Stefan Munteanu
Crystal Nakatsu
Apostol Natsev
Matteo Negri
Michael Oakes
Octavian Popescu
Oana Postolache
Christoph Reichenbach
Francisco Ribadas Peña
German Rigau
Tarek Sherif
Radu Soricut
Chris Stokoe
Rajen Subba
Hristo Tanev
Martin Thomas
Jesus Vilares Ferro
Zhuli Xie

Impreso en los Talleres Gráficos
de la Dirección de Publicaciones
del Instituto Politécnico Nacional
Tresguerras 27, Centro Histórico, México, D.F.
Febrero de 2006.
Printing 500 / Edición 500 ejemplares.

Natural Language Processing is an area of Artificial Intelligence aimed at development of computer programs capable of human-like activities related to written or spoken human language, such as English or Spanish.

This volume includes 19 original research papers by authors from 16 different countries on the following thematic areas of Natural Language Processing theory and applications:

- Lexical Resources
- Morphology and syntax
- Word Sense Disambiguation and Semantics
- Speech Processing
- Information Retrieval
- Question Answering and Text Summarization
- Computer-Assisted Language Learning

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 of computer science.

ISSN: 1665-9899

INSTITUTO POLITÉCNICO NACIONAL
"La Técnica al Servicio de la Patria"