EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

Instituto Politécnico Nacional
"La Técnica al Servicio de la Patria"

Research in Computing Science

ISSN: 1870-4069

Vol. 147 No. 1
January 2018

Research in Computing Science

Research in Computing Science, Año 17, Volumen 147, No. 1, enero de 2018, es una publicación mensual, editada por el Instituto Politécnico Nacional, a través del Centro de Investigación en Computación. Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, Ciudad de México, Tel. 57 29 60 00, ext. 56571. https://www.rcs.cic.ipn.mx. Editor responsable: Dr. Grigori Sidorov. Reserva de Derechos al Uso Exclusivo del Título No. 04-2019-082310242100-203. ISSN: en trámite, ambos otorgados por el Instituto Politécnico Nacional de Derecho de Autor. Responsable de la última actualización de este número: el Centro de Investigación en Computación, Dr. Grigori Sidorov, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738. Fecha de última modificación 01 de enero de 2018.

Research in Computing Science, year 17, Volume 147, No. 1, January 2018, is published monthly by the Center for Computing Research of IPN.

Volume 147(1)

Advances in Natural Language Processing

Alexander Gelbukh (ed.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2018

ISSN: 1870-4069

Formerly ISSN: 1665-9899

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

Indexed in LATINDEX, DBLP and Periodica

Electronic edition

ISSN 1870-4069

Table of Contents

Page

ISSN 1870-4069

Named Entity Recognition in Islam-related Russian Tweets

Valeria Mozharova, Natalia Loukachevitch

Lomonosov Moscow State University,
Russia

joinmek@rambler.ru, louk_nat@mail.ru

Abstract. The paper describes an approach to creating a domain-specific tweet collection written by users frequently discussing Islam-related issues in Russian. We use this collection to study specific features of named entity recognition on Twitter. We found that in contrast to tweets collected randomly, our tweet collection contains relatively small number of spelling errors or strange word shortenings. Specific difficulties of our collection for named entity recognition (NER) include a large number of Arabic and other Eastern names and frequent use of ALL-CAPS spelling for emphasizing main words in messages. We studied the transfer of the NER model trained on a newswire collection to the created tweet collection and approaches to decrease the degradation of the model because of the transfer. We found that for our specialized text collection, the most improvement was based on normalized word capitalization. Two-stage approaches to named entity recognition and word2vec-based clustering were also useful for our task.

Keywords: Islam russian, NER model, russian tweets, normalization of tweets.

1 Introduction

Available electronic texts in social media allows the study of the differences in language or style for specific communities [10, 16], or the dependence of the language on social and demographic characteristics of users [11, 23].

In Russia, one of the prominent groups of population are Muslims; the number of Islam believers is estimated as 10-30% of the total population. To study the language and the thematic interests of Muslims in Twitter, we created a specialized tweet collection. We begin our work with this collection from named entity recognition (NER) on Islam-related tweets.

Most studies of NER have been carried out on news collections and shown high quality of named entity extraction. However, the transfer of NER recognizers to other genres of texts usually demonstrates a significant decrease in the performance.

In particular, application of general NER recognizers designed for or trained on news collections can show the decrease in performance of up to 50% or more on informal texts published on social media platforms such as Twitter or Facebook [3, 6, 7, 8, 20].

In this paper we consider the transfer of a Russian NER recognizer trained on news texts to extracting names from Twitter messages. Our tweet collection is specialized, it is gathered from messages of those users who discuss issues related to Islam in their posts in contrast to other studies where Twitter collections are formed with random sampling of Twitter messages.

This allows us to reveal specific features of the tweet language of Islam-oriented and other similar communities. We consider the transfer of a CRF-based NER recognizer from a news data to the tweet collection and approaches to decrease the degradation of the model due to the transfer.

2 Related Work

2.1 Named Entity Recognition for Twitter

It is known that extraction of names from Twitter messages is a much more difficult task than from other genres of texts because of their shortness and informal character. In [6] the authors review the problems and approaches to named entity recognition and entity linking for tweets. They write that the tweet content is noisy because of incorrect spelling, irregular capitalization, and unusual abbreviations. In their experiments, the main sources of mistakes in named entity recognition in tweets were violations in capitalization, especially a large number of names written in lower case. They studied automatic normalization of tweets including spelling and capitalization correction and reported that in their investigation the normalization slightly improved the performance in NER for tweets.

In [19] the authors write that due to unreliable capitalization in tweets, common nouns are often misclassified as proper nouns, and vice versa. Some tweets contain all lowercase words (8%), whereas others are in ALL CAPS (0.6%). In addition to differences in vocabulary, the grammar of tweets differs from news text, for example, tweets often start with a verb. In their experiments the supervised approach was used to predict correct capitalization of words. The set of features included: the fraction of words in the tweet which are capitalized, the fraction which appear in a dictionary of frequently lowercase/capitalized words but are not lowercase/capitalized in the tweet, the number of times the word 'I' appears lowercase and whether or not the first word in the tweet is capitalized.

To study NER on Twitter performed with several NER systems, the authors of [7] use crowdsourcing to annotate tweets for the NER task. They annotate all @usernames as PER (person name). Annotating tweets for their experiments, the authors of [19] chose not to annotate @usernames mentioned in tweets as entities because it is trivial to identify them using a simple regular expression, and they would only serve to inflate the performance statistics.

In [3] the authors study the transfer of their NER model from news texts to tweets. They create a training set consisting of 1000 tweets. They use a baseline NER model based on token and context features (wordform, lemma, capitalization, prefixes and suffixes) and enhance it with two unsupervised representations (Brown clusters and vector representations) based on a large collection of unannotated tweets.

Besides, they propose a technique to combine a relatively small Twitter training set and larger newswire training data. They report that two unsupervised representations work together better than alone, and the combination of training sets further improves the performance of their NER system.

2.2 Named Entity Recognition in Russian

In Russian there is a tradition of engineering approaches to the named entity recognition task [12, 13, 22]. Machine-learning approaches for Russian NER usually employ the CRF machine learning method.

In [1] the authors presented the results of the CRF-based method on several tasks, including named entity recognition. The experiments were carried out on their own Russian text corpus, which contained 71,000 sentences. They used only n-grams and orthographic features of tokens without utilizing any knowledge-based features. They achieved 89.89% of F-score on three named entity types: persons (93.15%), geographical objects (92.7%), and organizations (83.83%).

In [17] the experiments utilized the open Russian text collection "Persons-600"[1] for the person name recognition task. The CRF-based classifier employed such features as token features, context features, and the features based on knowledge about persons (roles, professions, posts, and other). They achieved 88.32% of F-score on person names.

In [9] the experiments were carried out on the Russian text collection, which contained 97 documents. The authors used two approaches for the named entity recognition: knowledge-based and CRF-based approach. In the machine learning framework they utilized such features as the token features and the knowledge features based on word clustering (LDA topics [4], Brown clusters [2], Clark clusters [5]). They achieved 75.05% of F-score on two named entity types: persons (84.84%) and organizations (71.31%).

In 2016 the FactRuEval competition for the Russian language was organized. The FactRuEval tasks included recognition of names in Russian news texts, recognition of specific attributes of names (family name, first name, etc), and extraction of several types of facts [21]. The named entity recognition task on tweets has not been studied before for Russian. Also, the dependence of NER performance on the language of specific Twitter user communities has not been researched yet.

3 Text Collections

3.1 News Text Collection

We study the transfer of CRF-based NER classifier trained on newswire data to the tweet collection. For training our system, we chose open Russian text collection "Persons-1000", which contains 1000 news documents labeled with three types of named entities: persons, organizations and locations[2].

[1] ai-center.botik.ru/Airec/index.php?option=com_content&view=article&id=27:persons-600&catid=15&Itemid=40

[2] labinform.ru/pub/named_entities/descr_ne.htm

Table 1. The quantitative characteristics of the labeled named entities in text collections.

Type	News collection	Twitter collection
PER	10623	1546
ORG	8541	1144
LOC	7244	2836
OVERALL	26408	5526

The labeling rules are detailed in [15]. The counts of each named entity type in the collection are listed in Table 1.

3.2 Tweet Text Collection

We are interested in study of the language of Islam-related Twitter users in Russian. To extract tweets from users discussing Islam-related issues, we created a list of 2700 Islam terms. Then we extracted Russian tweets mentioning these terms using Search Twitter API, got users' accounts containing extracted tweets and ordered the accounts in the decreased number of extracted tweets from these accounts. We found that a lot of words from our list practically are not mentioned in tweets, other words (for example, "mosque" or "muslim") are often used by very different people, not only Muslims.

After studying tweets from the extracted accounts, we created a very small list of the main Islam words ("Allah", "Quran", "Prophet", in various forms of Russian morphology). We also added the names of several known islamist organizations to find their possible non-muslim proponents. Then we repeated the whole procedure of tweet and account extraction, and found that the extracted collections can be considered as an appropriate approximation of messages generated by Islam-related users.

We selected 100 users with the largest number of the extracted tweets, downloaded all their tweets and obtained the tweet collection consisting of 300 thousand tweets (further *FullTweetCollection*). Then we randomly extracted tweets from different users, removed non-Russian or senseless tweets and at last obtained the tweet collection of 4192 tweets (further *TestTweetCollection*).

The created collection contains messages with Quran quotes, religious and political argumentation, news-related messages mainly about Near and Middle East events (Syria, Iraq, Afghanistan etc) and Islamist organizations (Syrian opposition groups, ISIL, etc.) and also other types of messages (for example, advertisements).

The obtained collection was labeled similar to "Persons-1000". To annotate numerous mentions of Allah, we added the Deity type to the annotation scheme, but in the current study we consider the Deity type as a subtype of the Person type.

Analyzing the created collection from the point of view of NER difficulties we found that violations in capitalization mainly include ALL-CAPS words for the whole tweet and its fragment. Such capitalization is used for emphasizing important words in the text or words related to Allah as in the following example: За все потери, ОН дает нам большую награду" (*"For all the losses He gives us a great reward"*).

Also the tweets mention a lot of Eastern names of persons, organizations ("Фастаким кама умирт"(Fastakim Kama Umirt group), Джабхат фатх аш-Шам (Jabhat Fateh al-Sham)), or local places difficult for correct recognition.

The fraction of tweets with spelling mistakes, unusual shortenings is relatively low. We suppose that this is because the selected users are well-educated, they are professional writers in some sense, in most cases they are leaders of opinions, whose messages are retweeted by many other people. Therefore it is especially useful to study the specific features of their tweet language.

4 Description of NER Model

In our study we employ the baseline CRF-classifier that utilizes token features, context features, and lexicon features for NER. Then we consider the ways to improve the baseline model adapting it to the Twitter language. The adaptation techniques include the use of two stage-processing and unsupervised word clustering. Besides, we test the impact of tweet normalization on the NER performance.

4.1 Baseline Model

Before named entity recognition with CRF, tweets are processed with a morphological analyzer for determining the part of speech, gender, lemma, grammatical number, case and other characteristics of words. This information is used to form features of each word for classifying. In the baseline model we consider three types of features: local token features, context features, and features based on lexicons.

Token Features: The token features include:

- Token initial form (lemma),
- Number of symbols in a token,
- Letter case. If a token begins with a capital letter, and other letters are small then the value of this feature is ”BigSmall”. If all letters are capital then the value is ”BigBig”. If all letters are small then the value is ”SmallSmall”. In other cases the value is ”Fence”,
- Token type. The value of this feature for lexemes is the part of speech, for punctuation marks the value is the type of punctuation,
- Symbol ngrams. The presence of prefixes, suffixes and other ngrams from the predefined sets in a token.

Context-based Features The group of context features includes two feature types. The first type is local context features. It takes into account all mentioned token feature values of nearby words in two-word window to the right and to the left from the current word.

The second type is the bigram context feature. It contains information about the determined named entity type of the previous word. It helps to find named entity borders more precisely. For example, if the person second name is difficult for recognition, the presence of the first name before this word makes the classification easier.

Table 2. Vocabulary sizes.

Vocabulary	Size, objects	Clarification
Famous persons	31482	Famous people
First names	2773	First names
Surnames	66108	Surnames
Person roles	9935	Roles, posts
Verbs of informing	1729	Verbs that usually occur with persons
Companies	33380	Organization names
Company types	6774	Organization types
Media	3909	Media
Geography	8969	Geographical objects
Geographical adjectives	1739	Geographical adjectives
Usual words	58432	Frequent Russian words (nouns, verbs, adjectives)
Equipment	44094	Devices, equipment, tools

Features based on Lexicons: To improve the quality of recognition, we utilize special lexicons with lists of useful objects. An object can be a word or a phrase. The lexicons had been created before the current work and were not changed during the study.

To calculate the lexicon features, the system matches the text and lexicon entries. If a token is met in a matched lexicon entry then it obtains the lexicon feature value equal to the word length of the found entry. The use of the entry length as a feature helps to diminish the affect of lexical ambiguity.

For example, in the list of organizations there is “Apple” as the name of a company. But this word does not necessarily mean a company because it has the second “fruit” sense. In the opposite, if the phrase “Lomonosov Moscow State University” is found in the text, the probability of the organization sense is higher than in the first case. The lexicon feature containing the matched entry length helps the system to distinguish these two cases.

The biggest lexicons are listed in Table 2. The overall size of all vocabularies is more than 335 thousand entities. These lexicons were collected from several sources: phonebooks, Russian Wikipedia, RuThes thesaurus of the Russian language [14].

4.2 Adaptation of NER Model to Tweets

Unsupervised Word Clustering: In previous studies it was shown that unsupervised word clustering on the basis of a large text collection improves the NER performance [18, 15]. In our case we compare the impact of word clusters calculated on a large news collection and large tweet collection. For clustering we use the word2vec package[1]. It represents words with distributional vectors (word embeddings), computed with the neural network.

[1] github.com/dav/word2vec

The semantic similarity between two words is calculated as the cosine measure between two corresponding vectors. The package allows automatic clustering of words according to their calculated similarity. We used the cbow model with vector sizes equal to 300. Thus, each word has an additional feature – the number of a cluster in that it appears. The news collection utilized for clustering contains two million news documents. For tweet-based clustering we use a tweet collection consisting of randomly extracted Russian tweets and including 8.3 million tweets.

Two-stage Prediction: We suppose that for adapting a classifier to a text collection it can be useful to take into account the entities already labeled by the classifier and to memorize the named entity type statistics for future use.

At the first stage, the classifier extracts named entities. Then the system collects the class statistics determined at this stage for each word and uses it to form features of the second stage. After that, new features together with old ones participate in final classification. These statistics can be collected from the current text (the whole text or its part preceding to the word under analysis) or from a set of texts (collection statistics). In case of tweet processing, texts are small therefore only the collection statistics can be used. In our experiments this statistics can be obtained from the FullTweetCollection gathered from the selected user accounts or the labeled TestTweetCollection as described in Section 3.2.

For each word, the system finds all mentions of this word in the processed collection and counts frequencies of determined named entity types for this word. Using these frequencies for each entity type, the system creates additional features, which have one of three values: *no_one* (if the word has not been recognized as a named entity of the chosen type), *best* (if the word has been assigned to the chosen named entity type more then in 50% of cases), and *rare* (if the word has been assigned to the chosen named entity type less then in 50% of cases). For example, if the word "Russia" was met 500 times in a collection, and the classifier assigned it 200 times to organizations and 300 times to locations, then the values of the global statistics feature for the word "Russia" will be as following: *PER – no_one, ORG – rare, LOC – best*.

4.3 Normalization of Word Capitalization

As we found that in our tweet collection the share of misprints is not very high we did normalization only for word capitalization. The normalization was based on the large news collection described in Section 4.2. For each word in this collection, we counted how many times the word was written in letter case or capital case when it stands not in the beginning of a sentence. The more frequent case was considered as normal for this word. We considered the normalization in two variants:

- Variant A. All words in a tweet, except the first one, are changed to a normal form of capitalization,
- Variant B. All words in a tweet including the first one are changed to a normal form of capitalization.

We found that the variant B produces better results and later experimented only with this variant.

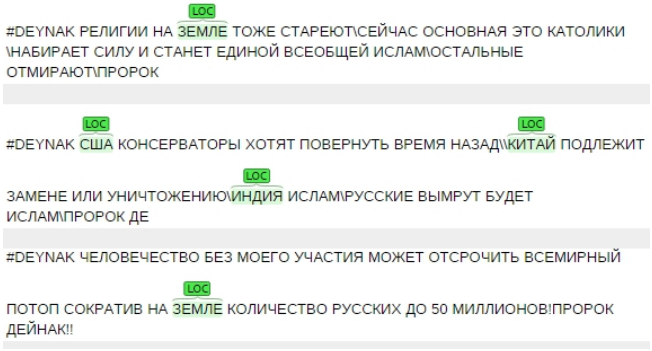


Fig. 1. Tweets before normalization.

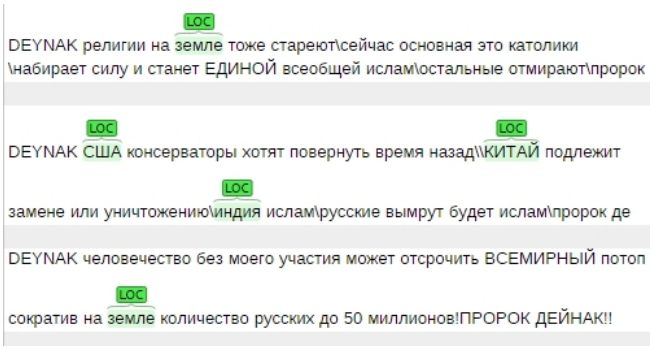


Fig. 2. Tweets after normalization.

Fig. 1 presents several tweets with the manual annotation before normalization. Fig. 2 shows the same tweets after normalization. Also the hashtag symbols were removed from a word if this word was found in the news collection to improve its matching with the lexicons.

5 Experiments

In preprocessing we removed mentioned user accounts with "@" and urls in the end of tweets. We consider these data as additional, as meta-information, from which we should not extract names. We train the described variants of our NER model on the news collection. Table 3 shows the results of named entity recognition on the "Persons-1000" collection (cross-validation 3:4).

It can be seen that our baseline model is quite good on the news collection and slightly improved after adding clustering features and the two-step approach. In this case the collection statistics is obtained from the same "Persons-1000" collection. Then we apply the trained model to the test tweet collection in initial capitalization and normalized capitalization.

Table 3. News collection NER performance.

Model	F-measure, %
Baseline	92.49
Baseline + News clusters	93.48
Baseline + News clusters + Collection statistics	93.53

Table 4. Tweet NER performance.

Model	F-measure, TestTweetCollection	F-measure, Normalized TestTweet-Collection
1) Baseline	64.44%	69.88%
2) Baseline + Collection statistics (TestTweetCollection)	64.99%	70.32%
3) Baseline + Collection statistics (FullTweetCollection)	65.78%	70.44%
4) Baseline + news clusters	66.03%	70.88%
5) Baseline + tweet clusters	66.08%	70.36%
6) Baseline + tweet and news clusters	66.23%	70.89%
7) (2) + tweet and news clusters	67.27%	71.20%
8) (3) + tweet and news clusters	66.46%	69.73%

Table 4 presents the performance of NER models trained on the ”Persons-1000” collection for the tweet data. One can see that all models significantly degrade on the tweet collection.

The normalization significantly improves the performance of NER (in contrast to other studies [6]). Word clustering and the collection statistics improve both NER for initial and normalized text collections. Their impact is larger than for the news collection (Table 3).

The combination of tweet and news clusters was better than only tweet clusters possibly because of the political and religious character of the gathered collection. In total, the NER performance improves more than 10% on the tweet data.

Analyzing the results generated with the best model on the normalized collection we can see still significant share of mistakes because of incorrectly normalized capitalization. The main subtypes of such problems include:

- Ambiguous words with different capitalization (”Earth”, ”Rose”) in different contexts.
- Words that should be capitalized in this specific collection. For example, ”Paradise” and ”Hell” seem to be specific entities in this genre of texts.
- Multiword expressions in that each word is usually written in letter case, but the whole multiword expression denotes a name and al least the first its word should be capitalized.

For example, the expression ”Московская область"(*Moscow region*) is normalized incorrectly because the word "московский" is written in letter case more frequently in the Russian news collection. However, ”Московская область" is the name of the location entity and should be capitalized.

6 Conclusion

The paper describes an approach to creating a domain-specific tweet collection written by users frequently discussing Islam-related issues in Russian. We use this collection to study specific features of named entity recognition on Twitter.

We found that in contrast to tweets collected randomly, our tweet collection contains relatively small number of spelling errors or strange word shortenings. Specific difficulties of our collection for named entity recognition include a large number of Arabic and other Eastern names (persons, locations, organizations) and frequent use of ALL-CAPS writing for emphasizing main words in messages.

We have studied the transfer of NER model trained on a newswire collection to the created tweet collection and approaches to decrease the degradation of the model because of the transfer. We found that for our specialized text collection, the most improvement was based on normalizing of word capitalization. Two-stage approaches to named entity recognition and word2vec-based clustering were also useful for our task. In future we plan to improve techniques of tweet normalization and study NER for tweets of followers of the selected users.

Acknowledgment. This work is partially supported by RFBR grant No. 16-29-09606.

References

1. Antonova, A., Soloviev, A.: Conditional random field models for the processing of russian. In: Computational Linguistics and Intellectual Technologies: Papers From the Annual Conference “Dialogue”, vol. 1, pp. 27–44 (2013)
2. Brown, P. F., Desouza, P. V., Mercer, R. L., Della-Pietra, V. J., Lai, J. C.: Class-based n-gram models of natural language. Computational Linguistics, vol. 18, no. 4, pp. 467–479 (1992)
3. Cherry, C., Guo, H.: The unreasonable effectiveness of word representations for Twitter named entity recognition. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 735–745 (2015) doi: 10.3115/v1/N15-1075
4. Chrupala, G.: Efficient induction of probabilistic word classes with LDA. In: Proceedings of the 5th International Joint Conference on Natural Language Processing, pp. 363–372 (2011)
5. Clark, A.: Combining distributional and morphological information for part of speech induction. In: 10th Conference of the European Chapter of the Association for Computational Linguistics, vol. 1, pp. 59–66 (2003) doi: 10.3115/1067807.1067817
6. Derczynski, L., Maynard, D., Rizzo, G., Van-Erp, M., Gorrell, G., Troncy, R., Petrak, J., Bontcheva, K.: Analysis of named entity recognition and linking for tweets. Information Processing and Management, Elsevier, vol. 51, no. 2, pp. 32–49 (2015) doi: 10.1016/j.ipm.2014.10.006

7. Finin, T., Murnane, W., Karandikar, A., Keller, N., Martineau, J., Dredze, M.: Annotating named entities in Twitter data with crowdsourcing. In: Proceedings of the NAACL Workshop on Creating Speech and Text Language Data With Amazon's Mechanical Turk, pp. 80–88 (2010)
8. Fromreide, H., Hovy, D., Søgaard, A.: Crowdsourcing and annotating NER for Twitter #drift. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation, pp. 2544–2547 (2014)
9. Gareev, R., Tkachenko, M., Solovyev, V., Simanovsky, A., Ivanov, V.: Introducing baselines for Russian named entity recognition. In: Lecture Notes in Computer Science, vol. 7816, pp. 329–342 (2013) doi: 10.1007/978-3-642-37247-6_27
10. Hidayatullah, A. F.: Language tweet characteristics of indonesian citizens. In: International Conference on Science and Technology (TICST), pp. 397–401 (2015) doi: 10.1109/TICST.2015.7369393
11. Hovy, D.: Demographic factors improve classification performance. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, vol. 1, pp. 752–762 (2015) doi: 10.3115/v1/P15-1073
12. Khoroshevsky, V. F.: Ontology driven multilingual information extraction and intelligent analytics. NATO Science for Peace and Security Series, vol. 27, pp. 239–262 (2010) doi: 10.3233/978-1-60750-611-9-239
13. Kuznetsov, I. P., Kozerenko, E. B., Kuznetsov, K. I., Timonina, N. O.: Intelligent system for entities extraction (ISEE) from natural language texts. In: Proceedings of the International Workshop on Conceptual Structures for Extracting Natural Language Semantics-Sense, vol. 9, pp. 17–25 (2009)
14. Loukachevitch, N., Dobrov, B. V.: RuThes linguistic ontology vs. russian wordnets. In: Proceedings of the seventh global wordnet conference, pp. 154–162 (2014)
15. Mozharova, V. A., Loukachevitch, N. V.: Combining knowledge and CRF-based approach to named entity recognition in russian. In: International Conference on Analysis of Images, Social Networks and Texts, Communications in Computer and Information Science, pp. 185–195 (2016) doi: 10.1007/978-3-319-52920-2_18
16. Paris, C., Thomas, P., Wan, S.: Differences in language and style between two social media communities. In: Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media, vol. 6 (2012) doi: 10.1609/icwsm.v6i1.14307
17. Podobryaev, A.: Person names recognition using CRF model. In: 15th All-Russian Scientific Conference Digital Libraries: Advanced Methods and Technologies, Digital Collections, pp. 255–258 (2013)
18. Ratinov, L., Roth, D.: Design challenges and misconceptions in named entity recognition. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning, pp. 147–155 (2009)
19. Ritter, A., Clark, S., Etzioni, O., Mausam: Named entity recognition in tweets: an experimental study. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 1524–1534 (2011)
20. Ritter, A., Etzioni, O., Clark, S., Mausam: Open domain event extraction from Twitter. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1104–1112 (2012) doi: 10.1145/2339530.2339704
21. Starostin, A., Bocharov, V., Alexeeva, S., Bodrova, A., Chuchunkov, A., Dzhumaev, S., Efimenko, I., Granovsky, D., Khoroshevsky, V., Krylova, I., Nikolaeva, M., Smurov, I., Toldova, S.: FactRuEval 2016: Evaluation of named entity recognition and fact extraction systems for Russian. In: Proceedings of the International Conference "Dialogue 2016", Computational Linguistics and Intellectual Technologies (2016)

22. Trofimov, I.: Person name recognition in news articles based on the persons-1000/1111-f collections. In: 16th All-Russian Scientific Conference Digital Libraries: Advanced Methods and Technologies, Digital Collections, pp. 217–221 (2014)
23. Yang, Y., Eisenstein, J.: Putting things in context: Community-specific embedding projections for sentiment analysis. Arxiv-Social Media Intelligence (2015)

ISSN 1870-4069

Proposed Novel Algorithm for Transliterating Arabic Terms into Arabizi

Nabeela Altrabsheh, Mazen El-Masri, Handay Mansour

University of Qatar,
College of Business and Economics
and Arabic Department,
Qatar

{nabeela, mazen.elmasri, hanadyma}@qu.edu.qa

Abstract. Arabizi is a new trend in social media where the person uses Latin characters to represent Arabic words. Arabic letters can be replaced with different symbols according to the dialects and preference. This creates a wide range of new vocabulary in sentiment lexicons. All the Arabizi literature focus has been on transliteration of Arabizi terms into Arabic terms. In this paper, we propose a new algorithm to transliterate Arabic terms into Arabizi. This allows a larger coverage of the different ways words are written in Arabizi, which will allow a more accurate analysis of the sentiment.

Keywords: Arabizi, arabic translator, arabizi literature.

1 Introduction

Arabizi is to write Arabic text using Latin characters. Arabizi can be used to write both Modern Standard Arabic (MSA) or Arabic dialects. Arabizi can be used for many reasons including technical limitations such as the keyboard not containing Arabic letters [2]. Another reason people like to use Arabizi is because it is free of errors and there are no typos as people can write words as they like [2]. People use Arabizi to write the letters as they pronounce them in real-life [1]. Arabic letters are pronounced differently according to the different dialects. These sounds could be expressed in Arabizi. Arabizi is dependent on the local dialect and differs from one country to another [9]. One example is the letter "ذ thal" which sometimes can be pronounced and written as 'z' in Egyptian dialect and 'th' in other dialects [3].

Another example is the letter "ق Qaf" which is pronounced 'Ga': Jordanian, 'Qa': Iraqi, 'Ka': Palestinian, and 'A': Lebanese [3, 9]. Another reason users type in Arabizi is that users are not familiar with typing in Arabic. Many studies show that people tend to write more in English these days [3, 9]. Therefore users find it difficult to switch between keyboards and find it easier to use Arabizi. Arabizi may also include some English abbreviations such as 'LOL' meaning laugh out loud. They may also include Arabic abbreviations such as 'ISA' meaning "In Sha Allah" and 'MSA' meaning "Ma Sha Allah".

Table 1. Arabizi letters.

Arabic	Arabizi				
ا	2	a	‘		
ؤ ئ أ ء	2				
ث	th	s			
ح	7	h			
ج	j	g			
خ	5	7	kh	x	
ذ	z	th	dh		
ش	sh	ch	$		
ص	s	9			
ض	d	9	dh		
ط	t	6			
ظ	th	6	dh	z	
ع	3	a	o		
غ	3	gh			
ق	8	q	g	2	9
و	w	o	ou		
ي	y	i	e		

Short vowels in Arabic consist of: fatha; a diagonal stroke written below the consonant which represents ‘a’, kasra; a diagonal stroke written below the consonant which represents ‘y’, ‘i’ or ‘e’, and damma; an comma shape written above the consonant which represents ‘u, ‘ou’, or ‘o’. When Arabic is typed online, short vowels are not usually written in the text. On the other hand, when writing Arabizi, users usually include the vowels. This makes it more easier for foreigners to read Arabic script [2]. Not writing the vowels in Arabic makes it difficult to transliterate text into Arabizi. Often a vowel changes the meaning of the word in Arabic, therefore, they are essential for transliteration. One example for this is the word حب which can be “Hubb” with the damma meaning love or “Habb” with the fatha meaning seeds.

Another issue with transliterating Arabizi to Arabic is that people tend to use both Arabizi and English in their sentences making it difficult to distinguish between Arabic and English words [3, 4, 2]. One example for this is the word ‘men’ which means ‘from’ in Arabic and is also a English word. There are only a few studies on Arabizi in the literature [3, 5, 1, 9]. Only one of the studies studied sentiment analysis on Arabizi [5]. All of research in Arabizi data was focused on transliterating Arabizi terms to Arabic and distinguishing between Arabizi and non Arabizi.

Table 2. Lexicons.

Database Name	Research	Size
ArabicSentimentLexicon	Mahyoub et al.	15110
Harvard Lexicon	Stone	1662
Arabic translation of Bing Liu's Lexicon	Mohammed et al.	6789
Arabic translation of MPQA Subjectivity Lexicon	Mohammed et al.	7619
Arabic translation of NRC Emoticon Lexicon	Mohammed et al.	26740

As there are many varieties of Arabizi letters that can be used, it is important for us to capture the different ways a single Arabic letter could be written in Arabizi. In this research, we propose an algorithm to transliterate Arabic terms into Arabizi. This will allow a larger coverage of Arabizi terms for a single Arabic term. Additionally, this algorithm will facilitate Arabizi sentiment analysis in social media and will lead to a more accurate analysis. One contribution of this paper is to create a Arabizi sentiment lexicon. Duwairi et al. [6] highlighted that there does not exist any Arabizi sentiment lexicon available. Related research is outlined in section 2. The research methodology including Arabizi letters, the data collected for this research is highlighted in section 3. In section 4, we present our proposed algorithm solution. Section 5 outlines the results and discussion, followed by the conclusions and future work in section 6.

2 Literature Review

Some researchers identified Arabizi from other languages such as [8, 4]. Others researched Arabizi transliteration such as [3, 1]. We will summarise these studies in the following paragraphs. Tobaili [8] research identifies Arabizi from multi-lingual data in Twitter. They collect two datasets from Twitter; one from Lebanon and the other from Egypt using geographic information. The data contained Arabic and non Arabic Tweets. Langdetect which is a library that detects one or more languages for a given sentence was used to detect the language.

From a sample of non Arabic Tweets, they extracted 3,707 Lebanese Arabizi and English tweets and 3,823 Egyptian Arabizi. Their results show that in Lebanon, the usage of Arabic and Non-Arabic tweets are nearly equal. On the other hand, in Egyptian tweets, Arabic is dominant. Arabizi tweets from Lebanon tend to be mixed with English and French, which is different from Egypt. In addition, Egyptians sometimes abbreviate Arabizi words in most cases by avoiding to write the vowels.

For example "2na w enta"- meaning you and I, which is written as "na w nta" in the Egyptian Arabizi text. They use SVM as a classifier and perform two experiments to distinguish between English and Arabizi tweets. The features included the languages detected by Langdetect, the language detected by Twitter API, and the count of word occurrences per tweet. The first experiment was with unbalanced data with the Arabizi being only 12% for Lebanon and 25% for Egypt.

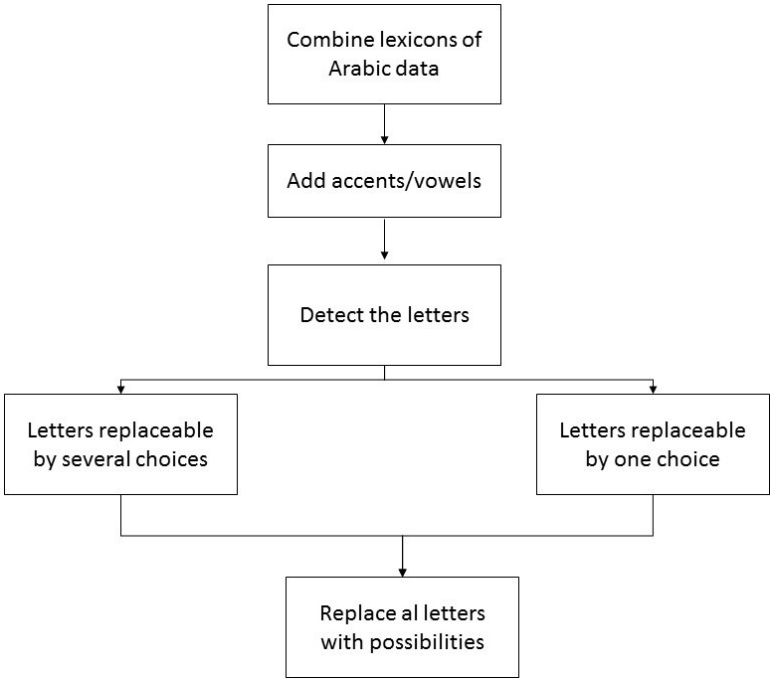


Fig. 1. Arabizi methodology.

This experiment resulted in a 93% accuracy. For the second experiment, undersampling was done and an accuracy of 97% was achieved. Darwish [4] presented methods to detect Arabizi from English text and transliterated Arabizi text into Arabic. For Arabizi detection, they used a sequence labeller which included word and character level features. They collected tweets using three keyword queries 'e7na' (we), '3shan' (because), and 'la2a' (no). They built a word list from tweets including 112 million Arabic tweets. An overall accuracy of 98.5% was achieved for Arabizi detection.

The second part of their research was to convert Arabizi to Arabic. A transliteration miner was trained to find the most likely Arabic word for the Arabizi word. Character transliteration probabilities and language modeling were used. For transliteration, they achieved a 88.7% accuracy. Bies et al. [3] research explored Arabizi-Arabic script transliteration. They collected SMS/Chat Corpus from 26 Egyptian participants. A number of 2,140 conversations and 475K words were collected. They also collected SMS from Broad Operational Language Translation program.

They automatically transliterated Arabizi into Arabic form. Annotators transliterated token by token, sentence by sentence and reviewed the corrected transliteration in full context. The second task was to correct spelling in the Arabic script transliteration to CODA (Conventional Orthography for Dialectal Arabic) standards. CODA minimizes differences between dialects and uses similarities and rules to combine them [7].

Table 3. Arabizi letter occurrences.

Letter	Occurrences
أ	25897
ث	1456
ح	5541
ج	4901
خ	2682
ذ	1027
ش	3592
ص	3249
ض	1932
ط	3332
ظ	709
ع	7587
غ	2500
ق	6122
ء	2542
و	11231
ي	16901

To correct spelling, different steps were taken. Firstly, the Egyptian words that had undergone sound changes such as the word 'mgfwl مقفول' which is sometimes written as 'm2fwl مأفول'. Some long vowels in words were written as short verbs such as 'qalt قالت' written as 'qlt قلت' . Some consonant letters are often mixed up with other consonants. For example the letters 'س and ص'.

Annotators corrected any mistakes of this type. They were also asked to correct errors from morphological ambiguities such as 'byemla بعملوا' and 'byemlh بيعمله'. Any word that was incorrectly segmented was corrected, such as 'fAl byt فال بيت' which should be 'fAlbyt فالبيت'.

Abbreviations such as 'INA' meaning 'In Sha Allah' were also changed to the full word form and typos in words were also dealt with. Lastly, words which were transliterated from English into Arabic such as "Have fun هف فن" were corrected. Al-Badrashiny et al. [1] presented a method to transliterate dialectal Egyptian Arabizi to Arabic by following CODA. The process they use is as follows: Arabizi sentences are input in their created system (3ARRIB).

Table 4. Letter occurrences in words.

Letter	Words
0	1410
1	8060
2	13986
3	10298
4	4685
5	2003
6	658
7	184
8	25
9	6
10	3

Preprocessing steps such lowercasing (de-capitalization), speech effects handling, and punctuation splitting are utilised. 3ARRIB creates a list of all possible transliterations for each word in the input sentence. This is by utilising a finite-state transducer that is trained on character-level alignment from Arabizi to Arabic text. They experiment with a morphological analyser for Egyptian Language (CALIMA) and a Language Model (LM). Several experiments were implemented. Consequently, they found that 3ARRIB with a 5-gram tokenized LM performed best, leading to an accuracy of 77.5% without normalisation and 79.1% with it.

3 Research Methodology

Yaghan [9], Attwa [2] Al-Badrashiny et al. [1] research covered the Arabic letters and their Arabizi alternatives. In this research, we combine these Arabizi letters and create a list with all the Arabizi replacements possible for each Arabic letter. An Arabic letter can be written in different English letters, numbers and characters. Most of the letters are easy to replace. However, there are some letters that are replaced differently according to the dialect, region and personal choice. These letters are illustrated in Table 1. There are 14929920 ($2\times2\times2\times2\times4\times3\times3\times2\times3\times2\times4\times3\times2\times5\times2\times3\times3$) possible combinations if all letters existed in the same word or phrase.

3.1 Data Collection

Data was collected from different online lexicons that are publicly available The total amount of words combined is 41318 Modern standard Arabic words. The database information is shown in Table 2. An amount of 16602 of these words were repeated, so we took the majority of polarity labels and dismissed the repeated instances. For example, if the word was negative twice and positive the once. We would keep one instance of the world with the negative polarity.

Some of the lexicons used scores; if the score was more than 0 it would be positive and less than 0 it would be negative. To combine all the tables we used only labels (i.e. positive, negative and neutral) and dismissed the scores.

3.2 Proposed Algorithm

In the beginning of this research, we attempted to use Darwish et al. [4] algorithm to convert the words from Arabizi to Arabic. Arabic vowels are fatha which is similar to 'a', kasra which is similar to 'e', and damma which is similar to 'o' [2]. However, we found when writing in Arabizi the vowels exist in the word. However in Arabic, the vowels are usually not typed into the writing and people read it intuitively [9]. We propose a new algorithm and follow the process illustrated in Figure 1.

1. Add diacritics to the words using an online tool. Miskhal tool uses a kaleidoscope linguistic approach based on phases of morphological analysis, grammar and semantic access[1].
2. Create a program that replaces each of the Arabic letters to all possible English letters, numbers, or characters.

One example to illustrate our method is the word: اتفاق. This can be written numerous ways as: etefaq, etefag, etefak, etefa8, etefa2, or etefa9.

4 Results and Discussion

We performed several analysis on our lexicon and found the following observations. The total number of words from the lexicon, taking into consideration all the possibilities of letters were 1426010. In the lexicon the most occurring letter was ا alef and the varieties of ء hamzas. The amount of letters occurrences (from Table 1) in the lexicon are presented in Table 3. We found that in most words only have two letters from Table 1. 1410 of the words in the lexicon did not contain any of the letters from Table 1. The amount of occurrences of letters in a single words are presented in Table 4. We found that the largest number of letters occurring together in the same word/phrase were 10 letters. The largest amount of possibilities for a single word/phrase was 38880=2×4×3×2×4×3×2×5×3×3 with the phrase:

السيكوباتي شخص مضطرب العقل "Alsekweyati is someone with mental problems".

This phrase included 10 letters ا، خ، ش، ص، ض، ط، ع، ق، و، ي.

Our proposed method is novel and can not be compared to any previous research. This research can facilitate in the creation of many new sentiment lexicons. It can also aid future systems for Arabic-Arabizi transliteration.

[1] tahadz.com/mishkal/index

Although the research has provided us with many possibilities for the same word. It is a weak approach as some of the words may not be used or written in real life. People tend to use certain combinations of letters, therefore, in future work we aim to find these combinations and narrow down the possibilities based on them. Also, another approach that can be used is to compare the lexicon with online words through social media websites (i.e. Twitter).

5 Conclusion and Future Work

In this research we explored transliterating Arabic terms into Arabizi. One of our main contributions of this research was creating an Arabizi lexicon which to the best of our knowledge has not been done before. In Arabic writing, mainly the vowels fatha, kasra and damma are not written, which makes it difficult to transliterate into Arabizi. Also, there are many alternative English letters, numbers and characters that can be written in Arabizi. We proposed a new method and algorithm for transliterating Arabic text into Arabizi. A lexicon was collected from different sources and transliterated into Arabizi. The number of words originally in the lexicon were 41318 Arabic words, the Arabizi transliteration for this lexicon was 1426010. We found that 17 letters had more than one possible replacement. We found that most words contained at two of these letters. For future work, we plan to create a search algorithm to find if there are any of the transliterated words from our system had previously been used online. This will allow us to discard Arabizi words that are not used or used less frequently.

Acknowledgments. This publication was made possible by the NPRP award [NPRP 7-1334-6-039 PR3] from the Qatar National Research Fund (a member of The Qatar Foundation). The statements made herein are solely the responsibility of the author[s].

References

1. Al-Badrashiny, M., Eskander, R., Habash, N., Rambow, O.: Automatic transliteration of romanized dialectal arabic. In: Proceedings of the 18th Conference on Computational Natural Language Learning. pp. 30–38 (2014)
2. Attwa, M.: Arabizi: A writing variety worth learning. Ph.D. thesis, The American University in Cairo, School of Humanities and Social Science (2012)
3. Bies, A., Song, Z., Maamouri, M., Grimes, S., Lee, H., Wright, J., Strassel, S., Habash, N., Eskander, R., Rambow, O.: Transliteration of arabizi into arabic orthography: Developing a parallel annotated arabizi-arabic script SMS/chat corpus. In: Proceedings of the Empirical Methods in Natural Language Processing Workshop on Arabic Natural Langauge Processing. pp. 93–103 (2014)
4. Darwish, K.: Arabizi detection and conversion to arabic. In: Proceedings of the Empirical Methods in Natural Language Processing Workshop on Arabic Natural Language Processing. pp. 217–224 (2014)
5. Darwish, K., Magdy, W., Mourad, A.: Language processing for arabic microblog retrieval. In: Proceedings of the 21st Association for Computing Machinery International Conference on Information and Knowledge Management. pp. 2427–2430 (2012)

6. Duwairi, R.M., Alfaqeh, M., Wardat, M., Alrabadi, A.: Sentiment analysis for arabizi text. In: International Conference on Information and Communication Systems. pp. 127–132 (2016)
7. Habash, N., Diab, M.T., Rambow, O.: Conventional orthography for dialectal arabic. In: Proceedings of the 18th International Conference on Language Resources and Evaluation. pp. 711–718 (2012)
8. Tobaili, T.: Arabizi identification in twitter data. In: Proceedings of the Association for Computational Linguistics Student Research Workshop. pp. 51–57 (2016)
9. Yaghan, M.A.: Arabizi: A contemporary style of arabic slang. Design Issues 24(2), 39–52 (2008)

ISSN 1870-4069

Hybrid Chunking for Turkish Combining Morphological and Semantic Features

Razieh Ehsani, Ercan Solak, Olcay Taner-Yıldız

Işık University,
Turkey

{razieh, ercan, olcaytaner}@isikun.edu.tr

Abstract. We use morphological features together with the semantic representations of words to solve chunking problem in Turkish. We separately train and tune word embeddings for semantic representations and conditional random fields for morphological features. We combine the two in a random forest.

Keywords: Fragmentation in trick, semantic and morphological model, conjunctions in turkish.

1 Introduction

Chunking is considered as an early form of parsing and defined as dividing a sentence into syntactically disjoint sets of meaningful word-groups or chunks [1]. An example of a sentence split up into chunks is shown below:

(1) [By midday] [NP the London market] [VP was in full retreat].

In Example (1), the chunks are represented as groups of words between square brackets, and the tags denote the type of the chunk.

Extracted chunks can be used as input in more complex natural language processing tasks, such as information retrieval, document summarization, question answering, statistical machine translation, etc. Compared to syntactic and/or dependency parsing, chunking is an easier problem. For this reason, in the earlier years of statistical natural language processing, many researchers put emphasis on the chunking problem [8].

In manually tagging a corpus, human annotators intuitively combine their linguistic knowledge with their competence of the language. In our approach, we try to capture these two sources of expertise and native knowledge about a language. We train a CRF with morphological features to learn the linguistic knowledge. For the competence, we train a neural network to represent the words as numeric vectors in an high dimensional space.

The paper is organized as follows: In Section 2, we give related work on chunking in Turkish and other languages. We describe our approach in Section 3. We report the experimental results in Section 4 and discuss them in Section 5. We conclude in Section 6.

2 Related Work

Ramshaw and Mitchell (1995) work [16] was one of the earliest works on chunking. They applied transformation-based learning approach for NP chunks on the data derived from Penn-Treebank. [9] applied support vector machines (SVM) to identify NP chunks. Using an ensemble of 8 SVM-based systems, they got 93% in terms of F-measure on CoNLL shared task data. The work in other languages, especially on less resourced and/or morphologically rich languages, is scarce. There are limited works on Korean [15, 13], Hindi [6], and Chinese [4, 18].

Kutlu's works [10] and [11] proposed the first Turkish NP chunker based on a dependency parser with handcrafted rules. NPs are divided into two sub-classes as main NPs (base NPs) and all NPs (including sub-NPs). Kahlout work [5] annotated verb chunks in METU-Sabancı Turkish dependency treebank [3] and replaced phrase chunks with constituent chunks. The remaining chunks are left as general chunks and only their boundaries are detected. The algorithm is based on conditional random fields (CRF) and achieves a best accuracy of 91.95 for verb chunks and 87.50 for general chunks.

3 Combining Nominal Features with Word Embeddings

For the modeling part, we train the semantic and morphological models separately and combine them in a random forest. Each part is tuned to maximize its performance in the chunking task when used in isolation.

3.1 Semantic Representation

We restrict ourselves to the semantics of a word as defined by the contexts in which it is used. To encode the context information we embed the words in real vectors using `word2vec` algorithm described in [14]. In order to customize the embeddings towards our task, we tuned the free parameters, window size and the embedding dimension of `word2vec` that maximize the performance in a partial evaluation of chunking.

3.2 Morphological Features

In order to train a learner for the morphological features in isolation, we use a CRF with a set of nominal features that are derived from a word. CRFs are known to perform well in sequential labeling tasks with nominal contextual features, [17].

For languages like Turkish where the syntactic functions of the words are often indicated by the case suffixes, morphological features become crucial for automatic labeling tasks. A particular example that is common in chunking is the genitive-possessive NP structure in Turkish. The genitive case marker and the possessive morpheme often also mark the boundaries of a NP. Even when the genitive possessive NP is composed of other NPs, anything between genitive marker and the possessive morpheme is often inside the outer NP.

Another set of case markers that are important in identifying propositional phrases in Turkish are the dative, locative and ablative markers, -A, -DA and -DAn.

Table 1. Morphological features.

Identifier	Definition
F0	Previous case
F1	Current surface form
F2	Previous surface form
F3	Next is ETOL, binary
F4	Previous POS
F5	Current is possessive, binary
F6	Previous is possessive, binary
F7	Current case
F8	Current is ETOL, binary

In English, starting boundaries of PPs are often marked with function words like "to," "at" and "from" These function words roughly correspond to the markers for the three noun cases just mentioned. As Turkish is a head final language, these three cases indicate the right boundary of a PP.

Many verbs in Turkish are formed by combining a noun with one of the two auxiliary verbs et- (do) and ol- (be). Examples are "yardım etmek" (to help), "kabul etmek" (to accept), "neden olmak" (to cause), "ait olmak" (to belong). For such cases, the auxiliary verb and noun preceding it must be chunked together. For surface forms we define the ETOL binary feature which indicates whether the root is et- or ol-.

The full set of features used in the isolated morphological chunking is given in Table 1. The contribution of the addition of each feature to the performance is given in the experiments in Table 1.

3.3 Combining Semantics and Morphology

Among the nominal features that we used in the CRF chunking were surface forms of the word itself and its surrounding words. Considering the effective vocabulary size of a morphologically rich language like Turkish, the surface forms in a training set make a rather sparse set of features. The same is to lesser extent true for the root forms of the words.

On the other hand, the morphological features have values from a small set, often binary. Instead of surface forms, we used their embedded representations as real vectors. This brings into the learner the semantics contained in the word embeddings. Besides, using vectors alleviate the problems of sparsity.

CRFs as commonly used for sequential labeling do not handle continuous valued features without a pre-processing such as binning. In order to directly combine the nominal features with continuous vector features, we use random forest.

4 Experiments

For training word2vec, we used the Milliyet corpus of Turkish which contains more than 5M sentences. For training and testing in the chunking task, we used the translated sentences of a portion of Penn Treebank with about 8K sentences [19].

Table 2. The average F-Scores for different embedding and window size combinations.

vector	**window**		
	1	2	3
5	68.1	66.2	65.7
10	69.8	69.3	67.9
20	**70.3**	69.3	68.7
50	69.4	69.4	68.2

Table 3. The average F-Scores for different contexts around the word to be tagged.

w_{t-2}	w_{t-1}	w_t	w_{t+1}	w_{t+2}	**F-Score**
		✓	✓	✓	73
	✓	✓	✓		**84.6**
✓	✓	✓			81.8
		✓	✓		74
	✓	✓			82.7
		✓			71.70

The first 90% is used for training and the rest for testing. We generated the chunk labels automatically using the root label of the largest subtree under the tree root.

We experimented with two sets of chunk labels. The simple set B, I only identifies the boundaries of the chunks. The larger set identifies the class of the chunks as well.

In order to find the most suitable window and vector sizes in `word2vec` for chunking task, we set up an SVM classifier that uses only the vector representation of word at t as the feature in deciding the chunk label at t. For the SVM and random forest implementations, we used the one provided by Weka, [7].

We used the simple label set $\{$B, I$\}$. The average F-scores are shown in Table 2. Comparing the F-scores, we chose a window size of 1 and an embedding size of 20. We used these parameters for the rest of the experiments.

Next, we determined the relevant context window for the surface forms of the words around the current word w_t. The largest window is $\{w_{t-2}, w_{t-1}, w_t, w_{t+1}, w_{t+2}\}$. We tested the random forest classifier with concatenated embeddings for window sizes of 1, 2 and 3 within this larger context. The weighted F-scores as percentages are given in Table 3. We see that the most relevant surface forms for chunking are $\{w_{t-1}, w_t, w_{t+1}\}$, one word to each side of the current word to be tagged.

To find the best morphological features for chunking, we used a CRF where for each set of candidate features, we also included the surface forms $\{w_{t-1}, w_t, w_{t+1}\}$. For a fast implementation of CRF, we used `wapiti`, [12]. Table 4 shows the incremental contribution of each feature to the overall performance. We used the floating feature selection algorithm [2] to determine the set of features in Table 1.

Finally, we combine the semantics and morphological features by using the best morphological feature set of CRF by replacing the surface forms with their embeddings.

Table 4. Feature selection for morphological chunking.

Feature set	Description	F-score
S0	F0	74.06
S1	S0 $\cup \{F1\}$	78.84
S2	S1 $\cup \{F2\}$	80.31
S3	S2 $\cup \{F3\}$	81.93
S4	S3 $\cup \{F4\}$	83.16
S5	S4 $\cup \{F5\}$	83.66
S6	S5 $\cup \{F6\}$	83.89
S7	S6 $\cup \{F7\}$	84.19
S8	S7 $\cup \{F8\}$	84.42

Table 5. Performance of the combined features for the set of simple chunking labels, B and I.

Label	F-score
B	84
I	88.9

The resulting set contains both numerical and nominal features. We used a random forest to combine these. The performance for the set of simple labels is given in Table 5.

The performance and the confusion matrix for the set of full labels are given in Table 6 and Table 7. In order to save space, we show a partial confusion matrix that contains only the most salient errors.

5 Discussion

We see that the highest performance is for CC which identifies conjunctions. This is somewhat expected as there are only a few conjunctions in Turkish and they can easily be identified through their surface forms.

The next best performance is for the start of verb groups (VG). Most verb groups in Turkish are just single words. The case of the previous word is an indicative feature in distinguishing the start of a verb group.

In English parsing, VP groups together the object NP and several PP and ADVP phrases under the same VP tree. However, for Turkish, such a grouping is too coarse. In generating our training and test data out of translated PennTreebank sentences, we automatically identified and extracted the verb under VP and chunked it as a verb group, VG. Then, we identified the remaining subtrees of VP tree as chunks with their own labels.

The confusion matrix in Table 7 highlights the distribution of errors in different chunks. Most salient error source is the wrong classification of words within a NP as either being inside a PP or S. PP errors must be due to the insufficiency of case marking features to differentiate between stand alone NPs and NPs within PPs.

Table 6. Performance of the combined features for the full set of chunking labels.

Label	F-score
B-CC	0.882
B-VG	0.738
B-NP	0.684
I-NP	0.632
B-ADVP	0.578
I-ADVP	0.491
I-PP	0.446
I-VG	0.411
I-S	0.395
B-PP	0.317
B-S	0.168
I-ADJP	0.095
B-ADJP	0
I-CC	0

Table 7. Confusion matrix for the full set of labels.

	Classified as								
	B-NP	I-NP	B-VG	B-PP	I-PP	I-ADVP	I-VG	B-S	I-S
I-NP	42	648	8	9	56	3	20	3	52
B-VG	25	21	282	6	2	0	12	2	16
B-PP	51	24	6	42	9	1	1	5	11
I-PP	10	**129**	1	6	127	4	3	0	31
I-ADVP	4	11	1	3	4	27	3	0	2
I-VG	5	72	41	3	14	4	77	0	19
B-S	56	32	3	9	3	0	0	15	10
I-S	28	**185**	27	14	32	10	10	5	162

Similarly, many sub-clauses in Turkish is bounded by genitive cases marking which is similar to genitive possessive NP constructions.

6 Conclusion

In this work, we proposed a new hybrid approach to Turkish chunking. We combined morphological features of the words together their semantic representations as real

valued vector embeddings. We trained a CRF and a random forrest separately to determine the best feature set and the best surface context around a word to be tagged. We combined the selected nominal features and surface embeddings in a random forest to achieve a hybrid chunker. For training CRF, we intentionally refrained from using the root forms of the words as it is not obvious how we would replace them with their vector embeddings because word embeddings were estimated from a corpus of surface forms. As a future study, we plan to device a method to meaningfully embed the roots and thus enrich the hybrid feature set.

References

1. Abney, S.: Parsing by chunks. Principle-Based Parsing, vol. 44, pp. 257–278 (1991) doi: 10.1007/978-94-011-3474-3_10
2. Alpaydın, E.: Introduction to machine learning. The MIT Press (2015)
3. Atalay, N. B., Oflazer, K., Say, B.: The annotation process in the Turkish treebank. In: Proceedings of 4th International Workshop on Linguistically Interpreted Corpora (2003)
4. Chen, W., Zhang, Y., Isahara, H.: An empirical study of Chinese chunking. In: Proceedings of the COLING/ACL on Main conference poster sessions, pp. 97–104 (2006)
5. El-Kahlout, I. D., Akin, A. A.: Turkish constituent chunking with morphological and contextual features. In: Computational Linguistics and Intelligent Text Processing, vol. 7816, pp. 270–281 (2013) doi: 10.1007/978-3-642-37247-6_22
6. Gune, H., Bapat, M., Khapra, M. M., Bhattacharyya, P.: Verbs are where all the action lies: Experiences of shallow parsing of a morphologically rich language. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, COLING '10, pp. 347–355 (2010)
7. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H.: The WEKA data mining software: An update. SIGKDD Explorations Newsletter, vol. 11, no. 1, pp. 10–18 (2009) doi: 10.1145/1656274.1656278
8. Jurafsky, D., Martin, J. H.: Speech and language processing: An introduction to natural language processing, computational linguistics and speech recognition. Prentice Hall series in artificial intelligence, (2009)
9. Kudo, T., Matsumoto, Y.: Chunking with support vector machines. In: Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies, pp. 1–8 (2001) doi: 10.3115/1073336.1073361
10. Kutlu, M.: Noun phrase chunker for Turkish using dependency parser. Master's thesis, Sabancı University (2010)
11. Kutlu, M., Cicekli, I.: Noun phrase chunking for turkish using a dependency parser. In: Lecture Notes in Electrical Engineering, ISCIS' 2015, pp. 381–391 (2014) doi: 10.1007/978-3-319-22635-4_35
12. Lavergne, T., Cappé, O., Yvon, F.: Practical very large scale CRFs. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 504–513 (2010) http://www.aclweb.org/anthology/P10-1052
13. Lee, Y. H., Kim, M. Y., Lee, J. H.: Chunking using conditional random fields in korean texts. In: Proceedings of the Second International Joint Conference on Natural Language Processing, pp. 155–164 (2005) doi: 10.1007/11562214_14
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of Workshop at ICLR (2013)

15. Park, S. B., Zhang, B. T.: Text chunking by combining hand-crafted rules and memory-based learning. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics, pp. 497–504 (2003) doi: 10.3115/1075096.1075159
16. Ramshaw, L. A., Marcus, M. P.: Text chunking using transformation-based learning. In: Natural Language Processing Using Very Large Corpora, vol. 11, pp. 157–176 (1995) doi: 10.1007/978-94-017-2390-9_10
17. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 134–141 (2003) doi: 10.3115/1073445.1073473
18. Sun, G. L., Huang, C. N., Wang, X. L., Xu, Z. M.: Chinese chunking based on maximum entropy Markov models. International Journal of Computational Linguistics & Chinese Language Processing, vol. 11, no. 2, pp. 115–136 (2006)
19. Yıldız, O. T., Solak, E., Görgün, O., Ehsani, R.: Constructing a Turkish-English parallel TreeBank. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, vol. 2, pp. 112–117 (2014) doi: 10.3115/v1/P14-2019

ISSN 1870-4069

An Approach to Information Retrieval from Scientific Documents

Divyanshu Bhardwaj[1], Partha Pakray[1], Alexander Gelbukh[2]

[1] NIT Mizoram,
Department of Computer Science and Engineering,
India

[2] Instituto Politecnico Nacional,
Centro de Investigación en Computación,
Mexico

{divbhardwaj42, parthapakray}@gmail.com
gelbukh@gelbukh.com

Abstract. Information retrieval (IR) is one the prominent fields o6f research in the present day. Despite this, mathematical information retrieval represents a callow niche that isn't delved into much. This paper describes the design and implementation of a Mathematical Search Engine using Apache Nutch and Apache Tomcat. The search engine designed is to be able to exhaustively search for mathematical equations and formula in scientific documents and research papers. Processed Wikipedia texts were used as inputs in order to probe the ability to search for specific mathematical entities among large volume of text distractors.

Keywords: Math search engine, retrieval of scientific documents, mathematical expressions, canonicalizer.

1 Introduction

Information retrieval (IR) is the activity of obtaining information resources relevant to an information need from a collection of information resources. In simple terms, it is the tracing and recovery of specific information from stored data. While the recognition and retrieval of textual information is a well versed discipline, retrieval of mathematical expressions is in its nascent stage.

Mathematical Information Retrieval is concerned with finding information in documents that include mathematics. Mathematical expressions are of utmost significance in scientific documents. They provide a means for the proper promulgation of scientific information. Despite the importance of math in technical documents, most search engines do not support users' access to mathematical formulae in target documents.

As such the need for a mathematical search engine capable of exhaustively searching documents for specific expressions and formulae is genuine.

In this paper, we delve into our attempt of performing information retrieval on scientific documents mainly looking to perform retrieval of mathematical expressions from Wikipedia articles. The challenge of working with mathematical entities is the ability of the expressions to assume different forms and yet imply the same thing. This makes absolute searching of mathematical expressions strenuous.

The rest of this paper is organised as follows, Section 2 describes the related works. Section 3 gives the dataset description. Section 4 gives the detailed description of the system architecture. Section 5 details the NTCIR Math Task. Section 6 provides an introduction to MathML. Section 7 lists out the experiments while section 8 discusses the results. Section 9 sums up the conclusion and lists out the future endeavours.

2 Related Works

Mathematical Information Retrieval has been an important point of research in the past few years. It has been a significant part of the annual NII Testbeds and Community for Information access Research (NTCIR) Conference[1]. In NTCIR-10, the NTCIR-Math Task was organized as two independent subtasks, first being Math Retrieval and the second Math Understanding. Larson et al. [9] linked a classic keyword content information retrieval method with bitmap indexing of math operators.

Schubotz et al. [17] applied a distributed data processing system that accesses data in a non-index format. This allowed for fast processing of batch queries. Kohlhase and Prodescu [6] implemented a web service that provides low-latency answers to unification queries over content MathML expressions. Topic et al. [19] employed an indexing scheme for mathematical expressions within an Apache Solr database.

This resulted is results being returned even if the variables were different or the structure was changed. Liska et al. [11] implemented a similarity search based on enhanced full text search. Hagino and Saito [4] used indices which would hold structure information of math expressions in order to build a partial match retrieval system for math formulae.

In NTCIR-11, the Math-2 Task required to develop a common workbench for mathematical formula search. Schubotz et al. [20] worked upon evaluation of Math Similarity factors based on the results from the pooling process. Gao et al. [2] employed a system aimed at searching for mathematical formulae based on both textual and spatial similarities. Pinto et al. [14] used a combination of a feature extracted sequence mechanism of the formulae and a sentence level representation of the text describing the formulae to model the collection.

Hambasan et al. [5] amended upon their previously built MathWebSearch with the effort directed on scalability, integration of keyword and formula search, and hit presentation issues. Kristianto et al. [7] apart from implementing an indexing scheme for mathematical expressions within an Apache Solr database proposed a reranking method making use of textual information and dependency graph. Ruzicka et al. [15] presented their second generation of scalable full text search engine Math Indexer and Searcher with an integrated canonicalizer.

[1] http://research.nii.ac.jp/ntcir

Table 1. Corpus statistics.

Corpus	Articles	Math Entities
ArXiv	105,120	∼60M
Wikipedia	319,689	592,443

Pattaniyil and Zanibbi [13] made use of two indices, viz. a Solr/Lucene based index for document text , and a MySQL index for math expressions. Lipani et al. [10] employed four indices, one for text, and three for mathematical formulas. NTCIR-12, concentrated upon the relevance assessments for formula search.

Gao et al. [3] developed a system which aimed to retrieve mathematical information based on keywords, and the structure and importance of formulae in a document. Also employed was a hybrid indexing and matching model in which both keyword and structure information of formulae were taken into consideration. Kristianto et al. [8] further improved their previously built indexing scheme for math expressions within an Apache Solr database with added features such as three levels of granularity for textual information, a method for extracting dependency relationships between math expressions, score normalization, cold-start weights, and unification.

Ruzicka et al. [16] revamped their previously implemented MIaS with the new features aimed primarily at further canonicalizing MathML input, structural unification of formulae for syntactic based similarity search, and query expansion. Davila et al. [1] made use of two indices, a Solr-based index for document text and a custom inverted index for math expressions.

Thanda et al. [18] implemented LDA and doc2vec's co- occurrence finding techniques, in addition to pattern and elastic search-based document ranking. Results from knowledge bases with different scoring mechanisms were merged using a nested Borda Count based technique.

3 Data

The corpus upon which the information retrieval was performed was the Wikipedia corpus of NTCIR-12 MathIR Task[2] . The corpus was divided into *math* articles containing <math>tags, and *text* articles that do not. We only considered the <math>tagged articles for our experiment as our main motive was to retrieve mathematical expressions from Wikipedia text. These consisted of 31,839 articles which was approximately 10% of the whole dataset. These articles in turn contained 580,068 formulae. The description of the data set of the NTCIR-12 MathIR task is given in table 1.

4 System Architecture

In order to implement the interface for the retrieval of mathematical equations and formula from documents, we set up a search engine which would be able to exhaustively search for them in the crawled and indexed database.

[2] http://ntcir-math.nii.ac.jp/introduction/

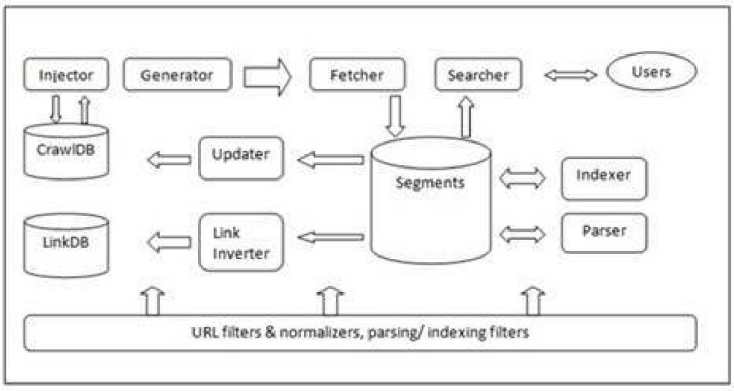


Fig. 1. System architecture of the Nutch based information retrieval system.

This search engine was entirely developed using Apache Nutch and run using Apache Tomcat. The graphical architecture of our Nutch implementation is given in Fig 1.

A prototype system was created by feeding the input data into Nutch. We implemented a generic Java code[3] to take the ids as the input to generate URL seeds. Next the injector injects the list of seed URLs into the *crawlDB*.

The generator then takes the list of seed URLs from *crawlDB*, forms fetch list and adds a crawl generate folder into the segments. These fetch lists are used by fetchers to fetch the raw content of the document. It is then stored in segments. Consequently the parser is called to parse the content of the document and parsed content is stored back in segments. The links are inverted in the link graph and stored in *LinkDB*.

This is followed by indexing of the terms present in segments and indices are updated in the segments. Following this information on the newly fetched documents is updated on the *crawlDB*. This crawl database is used as input in the implementation of the system. Queries are searched using this prototype system.

Results are obtained based on the ranking provided by Nutch. The architecture of Nutch as a web crawler can be interpreted from figure 2. The architecture of Nutch as a search engine can be interpreted from figure 3. We used the native ranking provided by Nutch for the development of the system. The ranking provided by Nutch may be explained using the following equation[4] :

$$\text{score}(\vec{q},\vec{d}) = \text{queryNorm}(\vec{q}) \times \text{coord}(\vec{q},\vec{d}) \times \text{norm}(t,\vec{d}) \times \sum_{t \epsilon \vec{d}} (tf(t) \times idf(t) \times t.\text{boost}(t.\text{field})), \quad (1)$$

where:

1. queryNorm() : indicates the normalization factor for the query,

[3] https://github.com/divyanshu42/IR/blob/master/ReadDirRecursevly.java
[4] Nutch And Lucene Framework- Arjun Atreya V RS-IITB

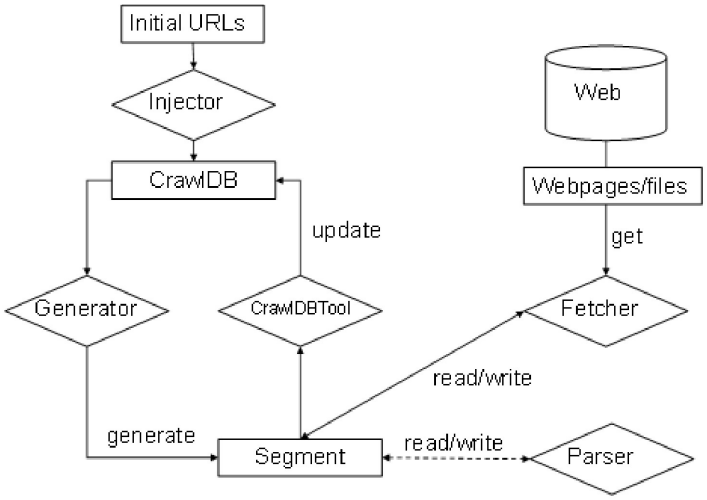


Fig. 2. Nutch as a web crawler.

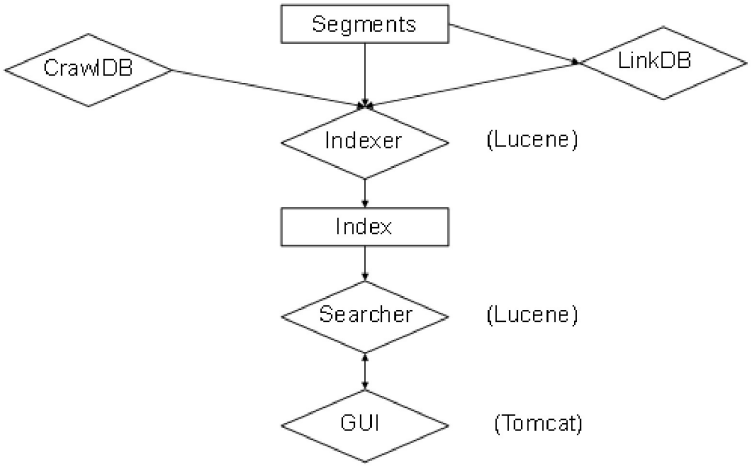


Fig. 3. Nutch as a complete search engine.

2. *coord() :* indicates how many query terms are present in the given document,
3. *norm() :* score indicating field based normalization factor,
4. *tf:* term frequency,
5. *idf:* inverse document frequency,
6. *t.boost() :* score indicating the importance of terms occurrence in a particular field.

5 NTCIR Math Task

NTCIR-12 Math Task is a shared task for retrieving mathematical information in documents in which queries are combinations of keywords and formula. Two corpora was used for the Math task. The first consisted of paragraphs from technical articles in the arXiv and the second consisted of complete articles from Wikipedia. For our system, we mainly utilised the Wikipedia corpus.

The arXiv dataset for NTCIR-12 MathIR consisted of scientific articles in English. Articles were converted from LaTeXto an HTML + MathML. The MathIR Wikipedia corpus contains articles from English Wikipedia converted into XHTML. 10% of the sampled articles contained explicit <math>tags that demarcated LaTeX. All articles with a <math>tag were included in the corpus.

The remaining 90% of the articles were sampled from Wikipedia articles that do not contain a *math* tag. These *text* articles functioned as distractors for keyword matching. Search topics were provided in a custom XML format. The topics consisted of the following items:
– Topic ID,
– Query (formula + keywords).

Given a set of queries, systems were to return a ranked list of search results for each of the arXiv and Wikipedia datasets.

6 MathML

Mathematical Markup Language (MathML)[5] is a mathematical markup language, an application of XML for describing mathematical notations and apprehending its structure and content. MathML is intended to reiterate mathematical and scientific content on the Web, and for other applications such as computer algebra systems, print typesetting, and voice synthesis.

MathML deals not only with the presentation but also the meaning of formula components (the latter part of MathML is known as "Content MathML"). Presentation MathML focuses on the display of an equation, and has about 30 elements. The elements' names all begin with m. A Presentation MathML expression is built up out of tokens that are combined using higher-level elements, which control their layout (there are also about 50 attributes, which mainly control fine details). Token elements generally only contain characters.

Content MathML focuses on the semantics, or meaning, of the expression rather than its layout. The <apply>element that represents function application. The function being applied is the first child element under <apply>, and its operands or parameters are the remaining child elements. Tokens such as identifiers and numbers are marked as *ci* and *cn*.

An example query from the NTCIR-12 math task in MathML is given in figure 4.

[5] https://www.w3.org/Math/

```
<topic>
 <num>NTCIR12-MathWiki-3</num>
 <query>
  <keyword id="w.0"> definition</keyword>
  <formula id="f.0">
   <math>
    <semantics>
     <mrow>
      <mi>a</mi>
      <mo>⊕</mo>
      <mi>b</mi>
     </mrow>
     <annotation-xml encoding="MathML-Content">
      <apply>
       <csymbol cd="latexml">direct-sum</csymbol>
       <ci>a</ci>
       <ci>b</ci>
      </apply>
     </annotation-xml>
     <annotation encoding="application/x-tex">a\oplus b</annotation>
    </semantics>
   </math>
  </formula>
 </query>
</topic>
```

Fig. 4. Example query in MathML to be retrieved by the system.

7 Experiment

We implemented a prototype system taking into account about 1GB of math tagged articles from the obtained data set. The steps involved in development of the prototype are mentioned in the underlying paragraph.

We start off by performing offline crawling for the data that is to be searched in. The crawl results in the formation of the crawl database known as *crawlDB.* The obtained crawl path is then added to the *nutch-default.xml* file in the *conf* directory.Next we build the nutch file using the *ant* and *ant war* commands.

Running these commands results in the the creation of the *war* file. This war file is then copied to apache tomcat folder *(-tomcat-(ver)/webapps/)*. We can now start our tomcat server. The apache tomcat server is started using the *catalina.sh start* command. In order to avoid the tedious task of running these commands each time, we created a shell script to automate it.[6] We now have a search engine up and running at the following address: *http://localhost:8080/nutch-(ver)/*

[6] https://github.com/divyanshu42/IR/blob/master/run.sh

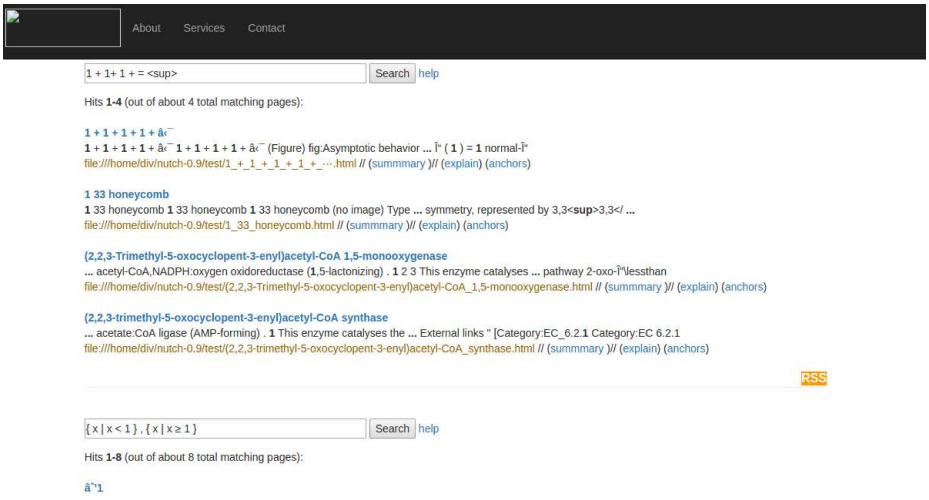


Fig. 5. Retrievals made by the system.

We implemented our search engine to search for a number of queries at the same time. This was achieved by modification of the search.jsp file which enabled us to supply a number of queries in a text file and search for them simultaneously.

During the experimentation, we tested the search engine for both mathematical expressions and text so as to compare performance of both the searches and also to be able to make sure that the search engine could be relied upon for textual searches.

8 Result

We obtained an extremely decent mathematical search engine which was capable of searching for mathematical expressions in the documents that comprised of the crawl database.

We were able to search for multiple queries in a single search which would prove to be extremely useful when searching for a number of math related equations and expressions at a given time. This has been shown in figure 5.

Upon analysis of the results we found that we obtained low precision in the retrieval for mathematical expression while we had relatively high accuracy for textual search.

For searching both math and text we implemented the same methodology, which only goes to draw attention to the challenges of working with mathematical expressions.

For instance a simple equation such as:

$$x^n + y^n = z^n. \tag{2}$$

Maybe represented in a number of different ways such as:

$$z^n = x^n + y^n, \tag{3}$$

$$or,$$

$$a^n + b^n = c^n. \tag{4}$$

Thus, the need of an efficient canonicalizer is genuine. The lack of such a canonicalizer meant that we could not retrieve all the different forms in which an expression may have been represented in some documents.

9 Conclusion and Future Works

In this paper we intended to delve into Information Retrieval (IR) in Scientific Documents by building a mathematical search engine which would be capable of searching exhaustively for mathematical expressions in scientific documents.

We made a lucid attempt at implementing an IR system and we obtained the expected results. While the results showed low precision for the mathematical expressions, this attempt proved to be a vital stride towards developing a fully functional and accurate retrieval system for mathematical entities in general. Now that we have a functional prototype capable of searching for mathematical expressions with decent accuracy and precision, our next endeavour would be working upon improving the precision of our system.

We plan to work upon a canonicalizer which would enhance the precision and give us better results while searching for mathematical entities. We would like to implement the canonicalizer and examine how it augments the precision and retrieval as a whole.

References

1. Davila, K., Zanibbi, R., Kane, A., Tompa, F. W.: The MCAT math retrieval system for NTCIR-10 math track. In: Proceedings of 12th NTCIR Conference (2016)
2. Gao, L., Wang, Y., Hao, L., Tang, Z.: ICST math retrieval system for NTCIR-11 math-2 task. In: Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-11, National Center of Sciences, pp. 9–12 (2014)
3. Gao, L., Yuan, K., Wang, Y., Jiang, Z., Tang, Z.: The math retrieval system of ICST for NTCIR-12 MathIR task. In: Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, pp. 318–322 (2016)
4. Hagino, H., Saito, H.: Partial-match retrieval with structure-reflected indices at the NTCIR-10 math task. In: Proceedings of NTCIR Conference on Evaluation of Information Access Technologies (2013)
5. Hambasan, R., Kohlhase, M., Prodescu, C.: Mathwebsearch at NTCIR-11. In: Proceedings of 11th NTCIR Conference, pp. 114–119 (2014)
6. Kohlhase, M., Prodescu, C.: MathWebSearch at NTCIR-10. In: Proceedings of 10th NTCIR Conference, pp. 675–679 (2013)
7. Kristianto, G. Y., Topić, G., Aizawa, A.: The MCAT math retrieval system for NTCIR-11 math track. In: Proceedings of 11th NTCIR Conference, pp. 120–126 (2014)
8. Kristianto, G. Y., Topić, G., Aizawa, A.: MCAT math retrieval system for NTCIR-12 MathIR task. In: Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, pp. 323–330 (2016)
9. Larson, R. R., Gey, F.: The abject failure of keyword IR for mathematics search: Berkeley at NTCIR-10 math. In: Proceedings of 10th NTCIR Conference, pp. 662–666 (2013)
10. Lipani, A., Andersson, L., Piroi, F., Lupu, M., Hanbury, A.: TUW-IMP at the NTCIR-11 math-2. In: Proceedings of 11th NTCIR Conference, pp. 143–146 (2014) doi: 10.13140/2.1.1127.8404

11. Liska, M., Sojka, P., Ruzicka, M.: Similarity search for mathematics: Masaryk university team at the NTCIR-10 math task. In: Proceedings of 10th NTCIR Conference, pp. 686–691 (2013)
12. Pakray, P., Sojka, P.: An architecture for scientific document retrieval using textual and math entailment modules. In: Proceedings of RASLAN 2014, pp. 107–117 (2014) doi: 10.13140/2.1.4036.2561
13. Pattaniyil, N., Zanibbi, R.: Combining TF-IDF text retrieval with an inverted index over symbol pairs in math expressions: The tangent math search engine at NTCIR 2014. In: Proceedings of 11th NTCIR Conference, pp. 135–142 (2014)
14. Pinto, J. M. G., Barthel, S., Balke, W. T.: QUALIBETA at the NTCIR-11 math 2 task: An attempt to query math collections. In: Proceedings of 11th NTCIR Conference, pp. 103–107 (2014)
15. Ruzicka, M., Sojka, P., Liska, M.: Math indexer and searcher under the hood: History and development of a winning strategy. In: Proceedings of 11th NTCIR Conference, pp. 127–134 (2014)
16. Ruzicka, M., Sojka, P., Liska, M.: Math indexer and searcher under the hood: Fine-tuning query expansion and unification strategies. In: Proceedings of 12th NTCIR Conference, pp. 331–337 (2016)
17. Schubotz, M., Leich, M., Markl, V.: Querying large collections of mathematical publications NTCIR10 math task. In: Proceedings of 10th NTCIR Conference, pp. 667–674 (2013)
18. Thanda, A., Agarwal, A., Singla, K., Prakash, A., Gupta, A.: A document retrieval system for math queries. In: Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, pp. 346–353 (2016)
19. Topic, G., Kristianto, G. Y., Nghiem, M. Q., Aizawa, A.: The MCAT math retrieval system for NTCIR-10 math track. In: Proceedings of the 10th NTCIR Conference on Evaluation of Information Access Technologies (2013)
20. Youssef, A., Schubotz, M., Markl, V., Cohl, H. S., Li, J. J.: Evaluation of similarity-measure factors for formulae based on the NTCIR-11 math task. In: Proceedings of 11th NTCIR Conference, pp. 108–113 (2014)

ISSN 1870-4069

A Supervised Learning for Assigning Categories to Medical Concepts and Contexts

Anupam Mondal[1], Erik Cambria[2],
Dipankar Das[1], Sivaji Bandyopadhyay[1]

[1] Jadavpur University, Kolkata,
Department of Computer Science and Engineering,
India

[2] Nanyang Technological University,
School of Computer Science and Engineering,
Singapore

anupam@sentic.net, dipankar.dipnil2005@gmail.com,
sbandyopadhyay@cse.jdvu.ac.in, cambria@ntu.edu.sg

Abstract. Category assignment of medical concepts is presented as an essential task in information extraction from a large number of available medical corpora in the form of unstructured and semi-structured. Besides, the category of medical concepts helps to identify the conceptual knowledge from the corpora, especially in healthcare. So, in the present task, we are motivated to design a category assignment system for the medical concepts and contexts which assists in understanding the subjective information of the medical corpus. The proposed system is able to identify five different types of categories for medical concepts (e.g., *diseases, drugs, symptoms, human_anatomy,* and *miscellaneous medical terms (MMT)*) and eleven types of context categories (e.g., *disease-symptom, disease-drug,* and *disease-MMT*). In order to design the system, we have distributed the task into three subtasks such as medical concept identification, concept category assignment, and category assignment for the context. Moreover, we have employed a domain-specific lexicon namely WordNet of Medical Event (WME 3.0) to design all three subtasks. The lexicon assists in recognizing the medical concepts and assigning the categories of these concepts using its provided various features as affinity score, gravity score, polarity scores, similar sentiment words, and sentiment. Thereafter, we have used two well-known supervised classifiers to build and validate the proposed both of the category assignment systems.

Keywords: Medical concepts, medical text categories, text classifier.

1 Introduction

In healthcare, category assignment of medical concepts and contexts are challenging due to the lack of involvement of domain-experts as well as unavailability of domain-specific lexicons and ontologies [27, 37].

In order to address these challenges, the researchers have developed several information extraction systems as Systematized Nomenclature of Medicine-Clinical Terms (SNOMED-CT), Unified Medical Language System (UMLS), and GENIA [4]. The systems help to recognize the medical concepts and their various conceptual features from unstructured corpora which assists in presenting structured corpus from unstructured corpora [18, 9].

On the other side, the researchers have designed various statistical and ontology-based approaches to identify the information automatically from the corpus [20]. The automated systems help the experts' like doctors and medical practitioners and non-experts as patients to describe the information related to medical concepts and contexts.

The concepts refer the key medical terms where the contexts represent the sentences of the corpus. For example, the medical concept "congenital_heart_disease" and medical context "A congenital heart defect (CHD) is a problem in the structure of the heart that is present at birth." respectively.

In the present paper, we are motivated to design an automated information extraction system namely categorization of medical concepts and contexts with the help of two well-known machine learning classifiers viz. Naïve Bayes and Logistic Regression [11]. Besides, we have prepared an experimental dataset and extracted features for medical concepts such as statistical, linguistic, and syntactical to build and validate both of the systems [36, 3].

We have also observed that a domain-specific lexicon is essential to extract the subjective and conceptual features of the medical concepts. Hence, we have employed previously developed a domain-specific lexicon viz. WordNet of Medical Event (WME 3.0) to design the proposed systems. The lexicon assists in recognizing the medical concepts in a context and extracting the features such as affinity score, category, gravity score, polarity score, Parts-Of-Speech (POS), sentiments, and similar sentiment words (SSW) for the medical concepts [26, 25].

We have also noticed the following difficulties exists to design the proposed categorization system. ***A.*** how to recognize the medical concepts (e.g., *breathing_problem*) and isolate the general concepts (e.g., organization) from the corpus? ***B.*** how to decide the set of medical categories and assign them to the medical concepts? This challenge is also presented as a classification task of medical concepts [28] viz.

A chronic cough is present either *symptom* or *disease* category. ***C.*** how to assign the set of categories for medical contexts and assign them? ***E.*** finally, how to prepare an experimental dataset to design and validate the proposed system? In order to address the mentioned challenges, we have used WME 3.0 lexicon which helps to recognize the medical concepts and isolate from non-medical concepts.

Thereafter, we have proposed five different types of categories for medical concepts such as *disease, symptom, drug, human_anatomy,* and *miscellaneous terms (MMT)* as unrecognized categories, suggested by medical practitioners after observing the experimental dataset. Besides, these categories assist in presenting eleven different pair-based categories for the medical context such as *disease-symptom*, *disease-drug*, *disease-human_anatomy*, *disease-MMT*, *symptom-drug*, *symptom-human_anatomy*,

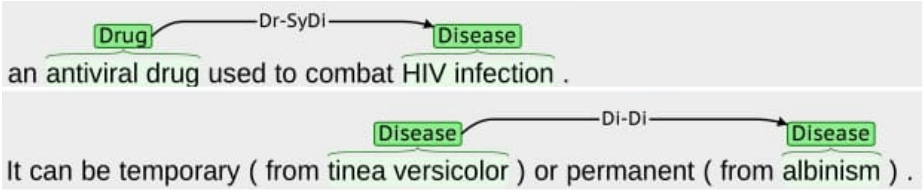


Fig. 1. A sample output of the proposed categorization system.

symptom-MMT, *human_anatomy-MMT*, *drug-human_anatomy*, *drug-MMT*, and *MMT-MMT*. Thereafter, we have employed Naïve Bayes and Logistic Regression classifiers along with various extracted features to develop the proposed categorization system.

The systems have provided F-score 0.81 and 0.86 for assigning the categories of medical concepts and contexts individually. Figure 1 shows a sample output of the proposed system.

We have observed that the categories of medical concepts and contexts facilitate to identify the subjective and conceptual information from the corpus. The system also helps to build various applications like medical concept network, annotation system, and recommendation system in healthcare services [22, 8, 9].

The rest of the paper is as follows. Section 2 presents the background of this work. Section 3 describes the preparation process of the experimental dataset and a brief overview WME lexicon. Section 4 discusses how we have developed both of the categorization systems such as concepts and contexts. Finally, the concluding remarks appear in Section 5.

2 Background

Information extraction from the biomedical textual corpus is demanding due to the lack of domain experts involvement and unavailability of structured corpora [21, 6]. In order to produce the structured corpus from the largely available semi-structured and unstructured corpora, the researchers have developed several ontologies and lexicons, which are also used for building automated information extraction system.

Few of the ontologies and lexicons are UMLS (Unified Medical Language System) and SNOMED-CT (Systematized Nomenclature of Medicine-Clinical Terms), MEN (Medical WordNet), and WME (WordNet of Medical Event) [33, 15, 25]. Primarily, these resources are looking for recognizing the conceptual as well as knowledge-based information from corpora [13, 5].

In the current research, we have focused on designing an automated categorization system using machine learning approaches in the presence of a domain-specific lexicons. So, in the following, we have discussed how the lexicons are taking a crucial role to develop the category assignment system for the medical concepts in healthcare.

Medical WordNet (MEN) lexicon built with two different sub-networks viz. Medical FactNet (MFN) and Medical BeliefNet (MBN), to evaluate consumer health reports [33].

The formal architecture of Princeton WordNet has been used to develop this lexicon [15]. On the other side, Mondal et al., [26, 25] have designed a domain-specific lexicon like WordNet of Medical Events (WME 3.0) to identify medical concepts and their various features. The features are affinity score, categories, gloss, gravity score, Parts-Of-Speech, polarity score, sentiment, and similar sentiment words. In this work, we have employed this lexicon to prepare and build the automated category assignment system for medical concepts and contexts.

Besides, we have noticed that the fundamental and broad classes such as category identification are still challenging due to insufficient number of domain experts involvement [23]. We have observed few of the category assignment research as follows: Yao, et al., [38] have extracted category-based relations as *cures, prevents,* and *side_effects*, which describe the distinctive nature from the biomedical text (medical papers) [2, 17].

Franzen et al., [16] have annotated Yapex corpus with 200 medical abstracts to assign the category such as *proteins*. Primarily, the ontology looks for extracting *protein-protein* interaction and *disease-treatment* relations from corpora under a BioText project [1, 30, 31, 19]. Eklund [12] developed an annotation system to extract the relations as *diseases* for *treatments* from the scientific medical corpus.

The literature survey helps to motivate to design the category assignment system for medical concepts and contexts both. Hence, we have used WME 3.0 lexicon and two well-known classifiers such as Naïve Bayes and Logistic Regression to prepare the experimental dataset, design, and validate the system. The concept categories show the subjective information for each of the medical concepts whereas context categories present the conceptual relation between each pair of medical concepts in a context.

3 Resource and Dataset

The present section describes how we employed WordNet of Medical Event (WME 3.0) lexicon to prepare the experimental dataset for the proposed categorization system.

*WordNet of Medical Event Lexicon:*A domain-specific lexicon is demanding to identify the conceptual information such as category or sentiment from the medical corpus [7]. So, we have borrowed the knowledge from the current version of WordNet of Medical Event lexicon namely WME 3.0 [26, 25, 24].

WME 3.0 lexicon was developed with an additional category feature of 10186 number of medical concepts along with previous two versions such as WME 1.0 and WME 2.0. WME 1.0 is able to identify the gloss (descriptive explanation), Parts-Of-Speech (POS), polarity score, and sentiment for 1654 number of medical concepts where WME 2.0 was added affinity score, gravity score, and Similar Sentiment Words (SSW) features for 6415 number of medical concepts.

The conventional WordNet[1], a preprocessed medical dictionary, and SenticNet[2] resources were applied to prepare all three versions of WME [10, 35]. In the following, we have discussed how the affinity score, gravity score, and category features help to design the proposed system.

[1] https://wordnet.princeton.edu/

[2] http://sentic.net/

```
<?xml version="3.0" encoding="UTF-8"?>
- <Medical Concepts>
 - <Concept>
     ……..
  </Concept>
 - <Concept>
    <Title>amnesia</Title>
   - <Properties>
      <POS>noun</POS>
      <Category>disease</Category>
      <Gloss>Loss of memory sometimes including the memory of
       personal identity due to brain injury, shock, fatigue, repression,
       or illness or sometimes induced by anesthesia.</Gloss>
        - <SSW and Affinity score>
                blackout (0.674)
                memory loss (0.534)
                stupor (0.429)
                fugue (0.345)
          </SSW and Affinity score>
      <Polarity score> - 0.375</Polarity score>
      <Gravity score>0.170</Gravity score>
      <Sentiment>negative</Sentiment>
    </Properties>
  </Concept>
 - <Concept>
     ……..
  </Concept>
</Medical Concepts>
```

Fig. 2. The current version of WordNet of Medical Event (WME 3.0) lexicon. The features considered in WME 3.0, affinity score, category, gloss, gravity score, Parts-Of-Speech (POS), polarity score, sentiment, and Similar Sentiment Words (SSW) are listed for an example medical concept ***amnesia***.

1. Affinity score indicates what extend two concepts are close to each other by measuring the number of common sentiment words (SSW) appeared for a pair of concepts within the range of 0 to 1.
2. Gravity score identifies sentiment-oriented relevance between medical concepts and their various glosses (descriptive explanations) and ranges from -1 to 1. While -1 suggests no relation and 1 indicates strong relations either positive or negative, which helps to identify a proper gloss for concepts.
3. Besides, the assigned categories such as *diseases, drugs, treatments, human_anatomy,* and *MMT* assist in extracting the subjective information of the concepts.

In the present paper, WME 3.0 lexicon has been used as a baseline to build a category assignment system. Figure 2 shows a sample output of WME 3.0 lexicon.

Experimental Dataset: A label dataset is essential to build and validate any information extraction system with machine learning classifiers. Hence, we have acquired and prepared an experimental dataset from two different resources such as one SemEval 2015 Task-6[3] and two MedicineNet[4].

[3] http://alt.qcri.org/semeval2015/task6/

[4] http://www.medicinenet.com/script/main/hp.asp

Table 1. A comparative statistics between two resources namely SemEval 2015 Task-6 and MedicineNet which are used to prepare the experiment dataset.

	SemEval 2015 Task-6	**MedicineNet**
Total number of contexts (both medical and non-medical)	4023	3329
Unique number of Medical contexts	2481	2583
Total number of concepts (only medical)	14665	16809
Unique number of Medical concepts	4485	4847
Average number of Medical concepts per contexts (respect to unique)	1.81	1.88
No. of mentioned categories	3	4

Table 2. A detail statistics of the experimental dataset.

Statistics of the labelled Experimental dataset				
		Experimental	**Training (40%)**	**Test (60%)**
Context Categories	*Disease-Symptom*	938	375	563
	Disease-Drug	713	285	428
	Disease-Human_anatomy	315	126	189
	Disease-MMT	535	214	321
	Symptom-Drug	108	43	65
	Symptom-Human_anatomy	78	31	47
	Symptom-MMT	135	54	81
	Human_anatomy-MMT	105	42	63
	Drug-Human_anatomy	285	114	171
	Drug-MMT	117	47	70
	MMT-MMT	1735	694	1041
Concept Categories	*Diseases*	2246	898	1348
	Symptoms	1204	482	722
	Drugs	2108	843	1265
	Human_anatomy	428	171	257
	MMT	3346	1338	2008

The experimental dataset contains 5064 and 9332 number of unique medical contexts and concepts individually after combining both of the resources. Table 1 presents a statistical comparison between these resources.

Besides, a group of medical practitioners helps to label both of the experimental datasets in the presence of WME 3.0 lexicon. Thereafter, we have split the datasets into two parts such as training (40%) and test (60%) dataset. The training dataset uses for building the systems where test dataset applies for validating them. Table 2 shows the distributions of both of the experimental datasets in details.

In the following sections, we have discussed how the prepared experimental dataset assists in building the categorization system for medical concepts and contexts.

4 Category Assignment Systems

The conceptual classes of medical concepts present the categories which help to identify the conceptual information from the corpus [32].

Both of the categories such as concepts and contexts of medical concepts assist in representing structured corpus from a large number of the unstructured and semi-structured corpora produced by the medical practitioners. Besides, these categories help to design various decision-making systems in healthcare services as annotation and recommendation etc [34, 29]. Hence, in the present work, we have recognized five different types of categories of medical concepts and their related eleven types of contexts categories.

The categories of medical concepts are *disease, drug, human_anatomy, symptom* and *MMT* where the categories of contexts are *disease-symptom*, *disease-drug*, *disease-human_anatomy*, *disease-MMT*, *symptom-drug*, *symptom-human_anatomy*, *symptom-MMT*, *human_anatomy-MMT*, *drug-human_anatomy*, *drug-MMT*, and *MMT-MMT*. We have proposed categories of the context as a pair-based due to the short length in the experimental dataset.

Thereafter, to assign the mentioned categories for both concepts as well as contexts, we have proposed two different categorization systems using machine learning classifiers individually. First categorization system is able to identify the subjective information from the corpus whereas the second system is able to extract the overall conceptual information from the corpus. In the following subsections discuss, how we have developed and validated both of the categorization systems in details.

4.1 Concept Categorization System

In order to design the proposed category assignment system for the medical concept, we have extracted various linguistic as well as knowledge-based features with two well-known supervised classifiers viz. Naïve Bayes and Logistic Regression. The features are POS, uni-gram, bi-gram, tri-gram, sentiment, and Similar Sentiment Words (SSW). To extract these features, we have employed WME 3.0 lexicon along with conventional WordNet, SentiWordNet, and SenticNet resources [14, 10].

Initially, we have extracted the mentioned features for the training dataset as mentioned in Section 3 and process it through two different classifiers individually. Thereafter, we have combined both of the classification modules to build the final category assignment system for medical concepts. The following steps discussed the system development in details.

- **Step-1:** Extract various features for the medical concepts of the training dataset using the mentioned resources (e.g., WME 3.0 and WordNet) and prepare a feature vector, F_V = [(1,0,1,1,1,1), (1,1,0,1,0,1), (1,1,0,1,0,0), ..., (1,1,1,1,0,1)].
- **Step-2:** Assign five different category label C_L = [1, 2, 2, ..., 4] to the feature vector F_V. The category labels are 1, 2, 3, 4, and 5 for *diseases, symptoms, drugs, human_anatomy*, and *MMT* individually.
- **Step-3:** The feature vector F_V and its corresponding label C_L have applied through two different classifiers such as Naïve-Bayes and Logistic regression and build two approaches as M_{NB} and M_{LR} respectively.
- **Step-4:** Thereafter, we have combined these approaches M_{NB} and M_{LR} with the union operator and built another approach ($M_{Combined}$) under the proposed module.

Table 3. An ablation study (as F-score) for assigning categories of medical concepts using three approaches viz. Naïve Bayes, logistic regression, and combined.

Features	M_{NB}			M_{LR}			$M_{Combined}$		
	Precision	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**
n-grams	0.79	0.91	0.85	0.80	0.89	0.84	0.81	0.83	0.85
n-grams + POS	0.82	0.89	0.85	0.83	0.87	0.85	0.84	0.88	0.86
Sentiment + SSW	0.75	0.83	0.79	0.77	0.85	0.81	0.78	0.84	0.81
POS + Sentiment + SSW	0.83	0.89	0.86	0.84	0.90	0.87	0.85	0.91	0.88
n-grams + Sentiment + SSW	0.84	0.88	0.86	0.83	0.89	0.86	0.86	0.91	0.88

Table 4. A comparative result analysis to assign categories of medical concepts using Naïve Bayes, logistic regression, and their combined approach.

	M_{NB}			M_{LR}			$M_{Combined}$		
	Precision	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**
diseases	0.85	0.97	0.91	0.85	0.97	0.91	0.86	0.98	0.92
symptoms	0.87	0.78	0.82	0.88	0.79	0.83	0.91	0.79	0.85
drugs	0.91	0.87	0.89	0.90	0.88	0.89	0.92	0.89	0.90
human_anatomy	0.67	0.90	0.77	0.70	0.88	0.78	0.67	0.91	0.77
MMT	0.79	0.84	0.81	0.81	0.84	0.82	0.86	0.81	0.83

For example, the proposed module is able to identify the medical concepts ***pneumonia*** and ***infection*** and their category as *disease* and *symptom* from the medical context "***Pneumonia*** *is frequently but not always due to* ***infection***".

4.2 Validation of Concept Categorization System

Thereafter, to validate the output of the proposed system, we have used the labeled test dataset as mentioned in Section 3. We have processed the test dataset with three modules viz. M_{NB}, M_{LR}, and $M_{Combined}$ consequently to calculate the accuracy of the final categorization system for medical concepts. Table 3 shows the importance of the applied features through an ablation study for the proposed system. We have also conducted a comparative result analysis for all three modules by calculating Precision, Recall, and F-score as shown in Table 4.

Finally, we have compared the category output of the proposed system along with WME 3.0 lexicon to prove the accuracy of the proposed system as shown in Table 5. We have also noticed that the coverage of the extracted categories of medical concepts is same with WME 3.0 lexicon due to use of the similar type of resources.

4.3 Context Categorization System

In addition, the category assignment system for the medical contexts is essential to determine the overall conceptual information for the corpus. So, we have proposed another categorization system on the top of the categorization system of medical concepts. The context categorization system is able to identify eleven types of pair-based categories for the medical corpus. Primarily these categories are derived after observing our experimental dataset by a group of medical practitioners.

Table 5. A statistical comparison of the coverage between the extracted categories from test dataset of the proposed system and WME 3.0 lexicon.

Distributions	**Test dataset**			**WME 3.0**
	Extracted	**Unique**	**Coverage (%)**	**Unique**
Disease	1321	1248	100	3641
Symptom	570	534	100	802
Drug	1126	1079	100	4196
Human_anatomy	234	206	100	227
MMT	1627	1185	85.07	1320

Table 6. A comparative result analysis to assign the categories of medical contexts using the proposed system.

	Test Dataset	**Extracted**	**Precision**	**Recall**	**F-score**
Disease-Symptom	563	545	0.95	0.98	0.96
Disease-Drug	428	397	0.89	0.96	0.92
Disease-Human_anatomy	189	165	0.80	0.92	0.86
Disease-MMT	321	278	0.82	0.95	0.88
Symptom-Drug	65	46	0.58	0.83	0.68
Symptom-Human_anatomy	47	32	0.62	0.91	0.74
Symptom-MMT	81	66	0.65	0.80	0.72
Human_anatomy-MMT	63	38	0.49	0.81	0.61
Drug-Human_anatomy	171	152	0.84	0.94	0.89
Drug-MMT	70	43	0.53	0.86	0.66
MMT-MMT	1041	971	0.77	0.83	0.80

Thereafter, to design the proposed categorization system for medical contexts, we have applied the following algorithm on training dataset in the presence of concept categorization system.

Step-1: Initially, we have assigned the categories of medical concepts in a context using concept categorization system. Each medical concept and its category presented as $Medi_C$ in a context.

Step-2: Consider the consecutive medical concepts and their category from the context, which is presented as Partial Context Category (PCC).

Step-2.1: If the consecutive pair of concept categories are same then PCC is:

$$PCC = Medi_{C1} \cap Medi_{C2}. \tag{1}$$

Step-2.2: Else PCC is:

$$PCC = Medi_{C1} \cup Medi_{C2}, \tag{2}$$

where $Medi_{C1}$ and $Medi_{C2}$ indicate two consecutive medical concepts and their categories.

Step-3: The extracted Partial Context Category (PCC) helps to assign the Final Context Category (FCC) as:

$$FCC = PCC_1 \cap PCC_2, \tag{3}$$

where PCC_1 and PCC_2 refer the partial context category in a context.

For example, the proposed system is able to assign *human_anatomy-MMT* category for the following medical context after identifying the medical concepts and their corresponding categories such as "***passage***" as *MMT*, "***air***" as *MMT*, "***supply***" as *MMT*, "***oxygen***" as *MMT*, "***lungs***" as *human_anatomy*, and "***body***" as *human_anatomy*.

"*The passage of air into and out of the lungs to supply the body with oxygen*".

4.4 Validation of Context Categorization System

In order to validate the proposed categorization system of medical contexts, we have used test dataset and calculated the accuracy in the form of F-score. Table 6 shows the extracted categories of medical contexts and a statistical comparison with manually labeled test dataset for medical context categories.

The presented result shows the importance of the proposed eleven categories of medical contexts to discover the overall conceptual information from the corpus. We have also observed that the category of the context provides an overall intuition whereas category of each concept of the context indicates specific fundamental class. Finally, we can conclude that the proposed systems may help to develop various medical applications like medical concept network, annotation system, and recommendation system in healthcare.

5 Conclusion and Future Scope

Category refers the broadest fundamental classes of any information extraction system especially in medical domain to identify the subjective information from the corpus. It also helps to represent structured corpus from the largely available unstructured and semi-structured corpora. So, in this paper, we are motivated to design the categorization system for medical concepts and contexts using supervised classifiers in the presence of a domain-specific lexicon namely WME 3.0.

The lexicon assists in preparing the experimental dataset and extracting various features from the dataset to build and validate the proposed categorization systems. The concept categories show the subjective information for each of the concepts whereas pair-based context categories indicate the overall conceptual information for the medical context. We are able to identify five types of concept categories as *diseases* and *symptoms* and eleven types of context categories like *disease-symptom* by the proposed systems individually. In order to build the category assignment system for medical concepts, we have amalgamated two classifiers namely Naïve Bayes and Logistic Regression. We have also conducted an ablation study to prove the accuracy of the system.

Thereafter, we have described how this system helps to design context categorization systems. In future, we will try to introduce more knowledge-based features to improve the accuracy of both of the systems. We will also focus on to design various applications to support expert and non-expert groups in their respective applications.

References

1. Abacha, A. B., Zweigenbaum, P.: Automatic extraction of semantic relations between medical entities: a rule based approach. Journal of Biomedical Semantics, vol. 2 (2011) doi: 10.1186/2041-1480-2-S5-S4
2. Abacha, A. B., Zweigenbaum, P.: A hybrid approach for the extraction of semantic relations from MEDLINE abstracts. In: International Conference on Intelligent Text Processing and Computational Linguistics, pp. 139–150 (2011) doi: 10.1007/978-3-642-19437-5_11
3. Basili, R., Pazienza, M. T., Vindigni, M.: Corpus-driven unsupervised learning of verb subcategorization frames. In: Congress of the Italian Association for Artificial Intelligence, pp. 159–170 (1997) doi: 10.1007/3-540-63576-9_105
4. Bodenreider, O.: The unified medical language system (UMLS): Integrating biomedical terminology. Nucleic Acids Research, vol. 32, pp. D267–D270 (2004) doi: 10.1093/nar/gkh061
5. Borthwick, A., Sterling, J., Agichtein, E., Grishman, R.: Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In: Proceedings of the Sixth Workshop on Very Large Corpora, vol. 182 (1998)
6. Brown, S. H., Elkin, P. L., Rosenbloom, S. T., Fielstein, E., Speroff, T.: eQuality for all: Extending automated quality measurement of free text clinical narratives. In: AMIA Annual Symposium Proceedings, vol. 2008, pp. 71–75 (2008)
7. Cambria, E.: Affective computing and sentiment analysis. IEEE Intelligent Systems, vol. 31, no. 2, pp. 102–107 (2016) doi: 10.1109/MIS.2016.31
8. Cambria, E., Hussain, A., Durrani, T., Havasi, C., Eckl, C., Munro, J.: Sentic computing for patient centered applications. In: IEEE 10th International Conference on Signal Processing Proceedings, pp. 1279–1282 (2010) doi: 10.1109/ICOSP.2010.5657072
9. Cambria, E., Hussain, A., Eckl, C.: Bridging the Gap between structured and unstructured healthcare data through semantics and sentics. In: Proceedings of the ACM Web Science Conference, pp. 1–4 (2011)
10. Cambria, E., Poria, S., Hazarika, D., Kwok, K.: SenticNet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings. In: The Thirty-Second AAAI Conference on Artificial Intelligence, vol. 32 (2018) doi: 10.1609/aaai.v32i1.11559
11. Chaturvedi, I., Ragusa, E., Gastaldo, P., Zunino, R., Cambria, E.: Bayesian network based extreme learning machine for subjectivity detection. Journal of The Franklin Institute, vol. 355, no. 4, pp. 1780–1797 (2018) doi: 10.1016/j.jfranklin.2017.06.007
12. Eklund, A. M.: Relational annotation of scientific medical corpora. In: LOUHI 2011 Third International Workshop on Health Document Text Mining and Information Analysis, pp. 27–34 (2011)
13. Embarek, M., Ferret, O.: Learning patterns for building resources about semantic relations in the medical domain. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation (2008)
14. Esuli, A., Sebastiani, F.: SENTIWORDNET: A publicly available lexical resource for opinion mining. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation, vol. 6, pp. 417–422 (2006)

15. Fellbaum, C.: WordNet: An electronic lexical database. The MIT Press (2000) doi: 10.7551/mitpress/7287.001.0001
16. Franzén, K., Eriksson, G., Olsson, F., Asker, L., Lidén, P., Cöster, J.: Protein names and how to find them. International Journal of Medical Informatics, vol. 67, no. 1, pp. 49–61 (2002) doi: 10.1016/s1386-5056(02)00052-7
17. Frunza, O., Inkpen, D.: Extraction of disease-treatment semantic relations from biomedical sentences. In: Proceedings of the 2010 Workshop on Biomedical Natural Language Processing, pp. 91–98 (2010)
18. Kim, Y., Riloff, E., Hurdle, J. F.: A study of concept extraction across different types of clinical notes. In: AMIA Annual Symposium Proceedings, vol. 2015, pp. 737–746 (2015)
19. Lee, C. H., Khoo, C., Na, J. C.: Automatic identification of treatment relations for medical ontology learning: An exploratory study. In: Proceedings of the Eighth International ISKO Conference, pp. 245–250 (2004)
20. Lee, C. H., Na, J. C., Khoo, C.: Ontology learning for medical digital libraries. In: International Conference on Asian Digital Libraries, vol. 2911, pp. 302–305 (2003) doi: 10.1007/978-3-540-24594-0_29
21. Meystre, S. M., Savova, G. K., Kipper-Schuler, K. C., Hurdle, J. F.: Extracting information from textual documents in the electronic health record: A review of recent research. Yearbook of Medical Informatics, vol. 17, no. 1, pp. 128–144 (2008) doi: 10.1055/s-0038-1638592
22. Mondal, A., Cambria, E., Das, D., Bandyopadhyay, S.: MediConceptNet: An affinity score based medical concept network. In: Proceedings of the Thirtieth International Florida Artificial Intelligence Research Society Conference, pp. 335–340 (2017)
23. Mondal, A., Cambria, E., Feraco, A., Das, D., Bandyopadhyay, S.: Auto-categorization of medical concepts and contexts. In: IEEE Symposium Series on Computational Intelligence, pp. 1–7 (2017) doi: 10.1109/SSCI.2017.8285253
24. Mondal, A., Chaturvedi, I., Das, D., Bajpai, R., Bandyopadhyay, S.: Lexical resource for medical events: A polarity based approach. In: 2015 IEEE International Conference on Data Mining Workshop, pp. 1302–1309 (2015) doi: 10.1109/ICDMW.2015.170
25. Mondal, A., Das, D., Cambria, E., Bandyopadhyay, S.: WME: Sense, polarity and affinity based concept resource for medical events. In: Proceedings of the Eighth Global WordNet Conference, pp. 243–248 (2016)
26. Mondal, A., Das, D., Cambria, E., Bandyopadhyay, S.: WME 3.0: An enhanced and validated lexicon of medical concepts. In: Proceedings of the Ninth Global WordNet Conference, pp. 10–16 (2018)
27. Mondal, A., Satapathy, R., Das, D., Bandyopadhyay, S.: A hybrid approach based sentiment extraction from medical context. In: Proceedings of the 4th Workshop on Sentiment Analysis where AI meets Psychology, pp. 35–40 (2016)
28. Neves, M.: An analysis on the entity annotations in biological corpora. F1000Research, (2014) doi: 10.12688/f1000research.3216.1
29. Pakhomov, S. V., Coden, A., Chute, C. G.: Developing a corpus of clinical notes manually annotated for part-of-speech. International journal of medical informatics, vol. 75, no. 6, pp. 418–429 (2006) doi: 10.1016/j.ijmedinf.2005.08.006
30. Rink, B., Harabagiu, S., Roberts, K.: Automatic extraction of relations between medical concepts in clinical texts. Journal of the American Medical Informatics Association, vol. 18, no. 5, pp. 594–600 (2011) doi: 10.1136/amiajnl-2011-000153
31. Rosario, B., Hearst, M. A.: Multi-way relation classification: Application to protein-protein interactions. In: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, pp. 732–739 (2005)

32. Shukla, R. S., Yadav, K. S., Abbas, S. T., Haseen, F.: An efficient mining of biomedical data from hypertext documents via NLP. In: Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014, vol. 327, pp. 651–658 (2015) doi: 10.1007/978-3-319-11933-5_73
33. Smith, B., Fellbaum, C.: Medical WordNet: a new methodology for the construction and validation of information resources for consumer health. In: Proceedings of the 20th international conference on Computational Linguistics, pp. 371–382 (2004) doi: 10.3115/1220355.1220409
34. South, B. R., Mowery, D., Suo, Y., Leng, J., Ferrández, O., Meystre, S. M., Chapman, W. W.: Evaluating the effects of machine pre-annotation and an interactive annotation interface on manual de-identification of clinical text. Journal of Biomedical Informatics, vol. 50, pp. 162–172 (2014) doi: 10.1016/j.jbi.2014.05.002
35. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. Computational linguistics, vol. 37, no. 2, pp. 267–307 (2011) doi: 10.1162/COLI_a_00049
36. Uzuner, Ö., South, B. R., Shen, S., DuVall, S. L.: 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. Journal of the American Medical Informatics Association, vol. 18, no. 5, pp. 552–556 (2011) doi: 10.1136/amiajnl-2011-000203
37. Xia, F., Yetisgen-Yildiz, M.: Clinical corpus annotation: Challenges and strategies. In: Proceedings of the Third Workshop on Building and Evaluating Resources for Biomedical Text Mining (BioTxtM'2012) in conjunction with the International Conference on Language Resources and Evaluation (2012)
38. Yao, L., Sun, C. J., Wang, X. L., Wang, X.: Relationship extraction from biomedical literature using maximum entropy based on rich features. In: 2010 International Conference on Machine Learning and Cybernetics, vol. 6, pp. 3358–3361 (2010) doi: 10.1109/ICMLC.2010.5580680

ISSN 1870-4069

Investigating Citation Linkage as an Information Retrieval Task

Hospice Houngbo, Robert E. Mercer

The University of Western Ontario,
Department of Computer Science,
Canada

hhoungbo@uwo.ca, mercer@csd.uwo.ca

Abstract. Scientists have a deluge of literature to consult in their research work. To navigate this overabundance of information, tools such as citation indexes help but do not indicate the precise passage in the paper that is being cited. In this study, we report our method for finding those sentences in a cited article that are the focus of a citation in a citing paper, a task we have called *citation linkage*. We first provide our guidelines for building a corpus annotated by a domain expert. The corpus consists of citing sentences and their cited articles. For this study the citing sentences deal with biochemistry methodology. All sentences in the cited article are annotated with six levels of relevance ranging from 0 (no relevance), to 5 (the annotator had the highest confidence that the sentence is relevant). We hypothesize that citation sentences when used as queries in a retrieval model should point to relevant sentences in the cited paper. To evaluate this hypothesis, a number of established retrieval algorithms using various document ranking methods are compared using information retrieval evaluation metrics. For each citation-paper linkage task, we compute Precision@k and Normalized Discounted Cumulative Gain, NDCG@k, where k is the number of sentences given non-zero relevance scores by the annotator. We found that 18 out of 22 citation linkage tasks have at least one sentence in the top k positions. The best Average NDCG is 49% and the best Average Precision is 50%.

Keywords: Linkage citations, syntagma, citations recovery algorithms.

1 Introduction

The writer of a research paper is required to place its contribution in its research context. This is often done by means of *citations*, which are instruments for connecting ideas in the research literature. The importance that citations play in the research literature has led to a variety of tools to assist researchers.

For instance, citation indexes, an idea conceived in 1964 [9], contain a subset of all of the citations in research articles. More recently, methods have been proposed to classify the purpose of a citation [12, 24]. Citation analysis-based bibliometrics are used to assess research and researcher importance [10, 11]. Potential uses of citations include multiple article summarization [6, 5, 21] and the tracking of scientific argumentation [18, 20] across multiple papers.

The purpose of a citation is typically to highlight a given aspect of the work described in the cited paper. However, citations in scientific writing refer to papers rather than the much smaller text span that is being referred to. Reducing citation targets from papers to a briefer text span such as a paragraph, a sentence, or a set of sentences, would be beneficial for some of the more complex applications mentioned above. This work, therefore, aims at investigating how to determine those sentences in a cited article that are the focus of a citation in a citing paper. We call this operation *citation linkage* and investigate it as an information retrieval problem.

We thus hypothesize that matching a citation sentence to relevant sentences in the cited paper can be compared to how search engines rank documents based on a user's specific query. In this study, each article is segmented into sentences and each sentence is compared to the citation sentence using a retrieval algorithm. The position of each candidate sentence is determined by ranking the similarity scores with regard to the citation sentence. The justification for such an hypothesis can be drawn from the necessary requirements for text retrieval experiments that the linkage task satisfies, such as:

- Information need: Retrieval from a given article, the sentences that match a specific citation sentence[1].
- Test collections: 22 articles, each containing relevant and non relevant sentences with multi level judgments.
- Evaluation methods: We can use ranked retrieval evaluation measures, such as Precision and Normalized Discounted Cumulative Gain (NDCG).

The rest of this paper is organized as follow. The next section reviews some related works. In Section 3, a detailed methodology of the citation linkage task is presented. In Section 4, the results are described. We conclude the paper with a summary and directions for future work.

2 Related Work

Finding the best linkage candidates (i.e., the sentences having more or less the same "contentful" meaning as a given citation sentence) will amount to computing the degree of similarity between the citation sentence and each sentence in the citing paper. Previous works on text similarity detection include paraphrase detection [8] and textual entailment [7]. Paraphrasing methods recognize, generate, or extract (e.g., from corpora) paraphrases, phrases, sentences or longer units of text that convey the same, or "almost" the same information.

Textual entailment methods, recognize, generate, or extract pairs (T, H) of natural language expressions, such that a human who reads (and trusts) T would infer that H is most likely also true [7].

[1] Text (or information) retrieval normally speaks of retrieving documents from a collection of documents. Our purpose here is to retrieve sentences from articles. Hence, we will simply substitute "sentence" for "document" and "article" for "collection" whenever the latter words are typically used in the text (or information) retrieval literature.

Both paraphrase detection and textual entailment have been combined in the SemEval2012 Semantic Textual Similarity shared task [2], consisting of finding similarity between sentences in a text pair (T1 and T2) and returning a similarity score and an optional confidence score. The authors participating in the tasks used a combination of lexical and syntactical approaches as well as machine learning approaches.

Nakov et al., [19] proposed the use of the text surrounding citations as tools for semantic interpretation of bioscience text. This work emphasized several different uses of citation sentences and showed that citation sentences are rich in domain specific concepts and terminology. The work in [19] aims at automatically extracting paraphrases of facts about a cited paper from multiple citations to it, with the eventual goal of using these sentences to automatically create summaries of the cited paper.

This is in line with Small's [23] notion of cited works as concept symbols, whereby a work may come to be repeatedly and consistently cited to represent a specific idea or topic, using descriptions that converge on an almost fixed terminology for that topic, such that the cited work eventually becomes synonymous with the topic. In [5], an attempt to match citation text and cited spans in biomedical literature proved to be a difficult task with limited performance.

3 Methodology

3.1 Building of a Citation Linkage Corpus

The text in which a citation occurs can span one or more sentences in the paper. In this study, this span is limited to one sentence, and the linkage task is assumed to be a sentence-level matching operation. An annotation guideline was defined to match a given citation sentence with candidate cited sentences based on the following criteria:

- To what extent can the person who reads a citation sentence taken in isolation be able to determine the candidate sentences that have been cited in a reference paper.
- Possible candidate sentences are chosen from the full article and presented chronologically as they appear in the article, thereby providing some context for the candidate sentence.
- For each sentence in the cited paper, a score will be given. This score will indicate the confidence that the annotator had in making his/her choice of candidate sentences. For those sentences not chosen as candidate sentences, a score of 0 is given indicating that the annotator is confident that there is no similarity in content with the citing sentence. For those sentences chosen as candidates, a score is given ranging from 1 (low confidence that similarity in content exists) to 5 (the annotator is confident that there is strong similarity between the candidate and citing sentences).

We have chosen the papers to annotate in such a way that they belong to a citation network that shows the links between cited and citing papers. We limit the current research to the biomedical domain and our final corpus is curated with papers from this domain. For this purpose, we use the BioMed Central's research articles corpus which is an open access corpus ideally suited for data mining research.

Table 1. Example of an annotation.

Citation Sentence	
We have been able to amplify 200 bp fragments of DNA obtained from Bouin's fixed and paraffin wax embedded tissues only after a specific restoration method to produce longer reconstructed DNA fragments.	
Candidate Sentences	**Rating**
To obtain longer stretches of DNA, a pre-PCR restoration treatment was required, by filling single strand breaks, followed by a vigorous denaturation step.	4
The development of this simple treatment allowed the analysis of longer fragments of DNA obtained from archival postmortem paraffin wax embedded tissues.	3
A partial restoration and reconstruction of DNA length in these cases is possible.	1
Here , we show that it is possible to analyze human postmortem paraffin wax embedded tissues amplifying a 287 bp sequence of apolipoprotein E (ApoE) and 291 bp of the prealbumin gene (TTR).	3
DNA was extracted from 6 m sections of paraffin wax embedded tissues.	1
The final sample of DNA was obtained by precipitation with ethanol using glycogen as the carrier.	1
DNA samples were incubated for one hour at 55C in 100 l of solution containing 10mM Tris/HCl (pH 8.3), 1.5 mM MgCl2, 2% Triton X-100, and 200M of each dNTP.	2
After this incubation, 1 U Taq DNA polymerase (Amersham) was added and DNA polymerisation was performed at 72C for 20 minutes.	2
The polymerase reaction restores the nicks after DNA rehybridisation, using the other strand as the template.	3
We have developed a method for amplifying longer DNA sequences, ranging up to 300 bases, from postmortem formalin fixed and paraffin wax embedded tissues, with no modification to the usual DNA extraction procedures.	5
Our restoration method is based on the fact that DNA degradation results from random single strand breaks and polymerase reaction restores the nicks, using the other DNA strand as a template.	2
Our restoration method is based on the fact that DNA degradation results from random single strand breaks and PCR restores the nicks, using the other DNA strand as a template.	2
The method proposed can be used to obtain longer amplification fragments of around 300 bp of DNA from normally extracted postmortem paraffin wax embedded tissues.	4

The richness of BioMed Central's XML format also makes the content especially suitable for information extraction and textual analysis. The citation sentences are limited to sentences that contain specific terminology that may represent a set of methods, tools or techniques used in scientific experiments and refer, we believe, to sentences of a similar kind in the cited papers.

Annotators' feedback has also been collected and points to the fact that candidate sentences were chosen by the annotators based on surface level similarity as well as non explicit factors such as background domain knowledge, and inferential deduction. Table 1 presents an example of an annotation produced by a human annotator (only the non-zero rated candidate sentences and their rating scores are presented).

Table 2. Number of candidate sentences per paper.

Paper	Score					Paper	Score				
#	**1**	**2**	**3**	**4**	**5**	#	**1**	**2**	**3**	**4**	**5**
1	4	3	6	4	2	12	3	0	1	1	0
2	0	4	4	3	5	13	1	0	0	1	0
3	0	0	7	1	4	14	8	0	0	3	1
4	1	0	1	1	0	15	4	2	2	5	1
5	6	1	1	9	1	16	2	0	3	2	1
6	0	0	1	1	1	17	1	6	1	5	2
7	1	0	1	0	2	18	2	1	2	0	1
8	3	0	3	1	0	19	2	2	0	0	0
9	1	1	1	0	0	20	0	0	0	2	1
10	0	0	3	2	3	21	0	0	1	0	2
11	2	17	4	3	5	22	3	5	0	4	1

The citation sentence is from the article: *DNA and RNA obtained from Bouin's fixed tissues*; https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1770606/. The candidate sentences are from the referenced article: *PCR analysis in archival postmortem tissues*; https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1187316/. All 22 annotations are available at https://github.com/hospice/linkagefiles. The annotation guidelines can be found in the first author's PhD thesis [16]. Table 2 shows the number of candidate sentences per paper.

3.2 Evaluation Metrics

When the sentences retrieved by the search techniques are evaluated on a binary relevance basis, each sentence is treated as being either relevant or irrelevant. *Precision* measures how many sentences are relevant to the query among the returned sentences.

In the computation of the output of these search operations, we are interested in the proportion of relevant results among the first k retrieved results. If the search technique returns r relevant sentences in the first k sentences that it finds, then *Precision*@k is:

$$\textit{Precision}@k = \frac{r}{k}.$$

Because each paper in our corpus has a different number of candidate sentences chosen to be relevant by the annotators, we have chosen k to be this number for each paper. In some cases, the degree to which a sentence satisfies the query needs to be taken into account during the evaluation process. When the relevance of sentences in the article can be captured with more than two classes, new measures are needed to capture these degrees of relevance of each retrieved sentence.

The overall score is obtained by combining "relevance" values and the position of the sentence in a ranked search result. Such measures include the Normalized Discounted Cumulative Gain (NDCG)[17]. The NDCG evaluation measure is computed to reflect the ideal position of the very relevant, the marginally relevant, as well as the non-relevant sentences in the ranking. The Normalized Discounted Cumulative Gain (NDCG) principle can be summarized as follows:

- It is applicable to multi-level judgments in a scale of $[1, r]$, $r \geq 2$.
- It measures the total utility of the top k sentences.
- The utility of a lower ranked sentence is discounted.
- The score is normalized to assure comparability across queries.

$$DCG@k = rel_1 + \sum_{i=2}^{k} \frac{rel_i}{\log_2 i}. \quad (1)$$

rel_1 is the graded relevance of the sentence at position 1.
rel_i is the graded relevance of the sentence at position i.

$$NDCG@k = \frac{DCG@k}{IDCG@k}, \quad (2)$$

where $IDCG@k$ is the Ideal Discounted Cumulative Gain for the k ordered results.

3.3 Ranking Models

We experiment with different ranking techniques such as BM25[22], Divergence From Randomness (DFR), Vector Space Model (VSM) [3], Information Based Similarity (IBS) [13], Language Model with Jelinek-Mercer smoothing (LMJ) [25] and Language Model with Dirichlet priors smoothing (LMD)[26]. The Lucene framework [4], which has the implementation of these techniques, is used for this purpose.

3.4 Index Creation

During the indexing process, each article is segmented into sentences, as the linkage is done at the individual sentence level. Sentences are transformed into fields of content (words). An analysis step is then performed to remove stop-words which are not required for the search operations. An index writer creates indexes as required.

3.5 Search Operation

A query expression is derived either from the unmodified citation text or a reformulation of it. Here we use two types of queries: the full unmodified citation sentence and the noun phrases found in the citation sentence. A query parser operation is performed after the query terms are analyzed for stop word removal. The query expression is then passed to the searcher module which returns ranked sentences based on a ranking model.

3.6 Experiments

Experiments can be divided into two main types:

1. Linkage operations using all sentences as potential candidate cited sentences.

2. Linkage operations using a reduced set of sentences as potential candidate cited sentences (for our experiments, here, these are sentences that have been automatically marked rhetorically as Method sentences [1, 14]).

For each type, we set up two experiments: In the first set of experiments, we use the words in the citation sentences as query terms. In the second set of experiments, we reduce the query terms to the noun phrases in the citation sentences. This second set is based on an hypothesis that noun phrases may contain the important query terms (the same hypothesis is found in [5]).

4 Results and Discussion

We use two evaluation metrics for our experiments, namely: *NDCG@*k and *Precision@*k. *NDCG@*k takes into account the multi-level utility score of each sentence, whereas *Precision@*k only takes into account the binary relevance of each sentence. For a given citation–article pair, the evaluation score will depend on the number of candidate sentences, which varies depending on the paper being considered.

Therefore the computation of an overall average score for all the papers is done after we compute the individual score for each paper, taking into account different numbers of candidate sentences. For a total number of N papers, if paper$_i$ has k_i candidates, we first compute *Precision@*k_i and *NDCG@*k_i for each paper$_i$, before computing the Average Precision, *Avg. Precision*, and the Average Normalized Discounted Cumulative Gain, *Avg. NDCG*:

$$\textit{Avg. Precision} = \frac{\sum_{i=1}^{N} \textit{Precision@}k_i}{N}, \tag{3}$$

And,

$$\textit{Avg. NDCG} = \frac{\sum_{i=1}^{N} \textit{NDCG@}k_i}{N}. \tag{4}$$

Table 3 shows the results for the experiments using the full citation sentence as query input and Table 4 shows the results of the experiments using noun phases extracted from the citation sentence as query input. Besides the average scores (Mean), three summary statistics for each retrieval model are presented: minimum, maximum, and median over all the papers. Four retrieval models, VSM, IBS, DFR, LMJ and LMD, show similar performance, with BM25 having much lower Mean values.

The Mean values vary between 0.2506 and 0.3412 for all sentences as possible candidates, and between 0.4064 and 0.4976 for Method sentences as possible candidates for *Precision@k*, and between 0.2514 and 0.3247 for all sentences as possible candidates and between 0.3958 and 0.4913 for Method sentences as possible candidates for the *NDCG@k* for the experiments using full citation sentences as query input. Similarly, the Mean values vary between 0.2362 and 0.3557 and between 0.3962 and 0.4988 for *Precision@k*, and between 0.2411 and 0.3590 and between 0.3702 and 0.4886 for the *NDCG@k* for the experiments using noun phrases as query input. Therefore, models with query reduction using noun phrases have slightly better performances.

Table 3. Evaluation: queries: citations.

Possible Candidates: All sentences in cited paper					Possible Candidates: Method sentences in cited paper				
Precision@k					***Precision@k***				
	Min.	**Median**	**Mean**	**Max.**		**Min.**	**Median**	**Mean**	**Max.**
IBS	0.0000	0.3229	0.3212	0.6667	IBS	0.0000	0.4808	0.4550	1.0000
VSM	0.0000	0.3542	0.3395	0.6667	VSM	0.0000	0.5000	0.4826	1.0000
BM25	0.0000	0.2283	0.2506	0.6667	BM25	0.0000	0.4476	0.4064	1.0000
DFR	0.0000	0.3333	0.2972	0.6667	DFR	0.0000	0.4727	0.4741	1.0000
LMJ	0.0000	0.3542	0.3412	0.6667	LMJ	0.0000	0.5000	0.4854	1.0000
LMD	0.0000	0.3205	0.3309	0.6667	LMD	0.0000	0.4919	0.4976	1.0000
NDCG@k					***NDCG@k***				
	Min.	**Median**	**Mean**	**Max.**		**Min.**	**Median**	**Mean**	**Max.**
IBS	0.0000	0.3293	0.3168	0.7751	IBS	0.0000	0.4830	0.4569	1.0000
VSM	0.0000	0.3271	0.3237	0.7751	VSM	0.0000	0.5197	0.4906	1.0000
BM25	0.0000	0.2984	0.2514	0.7654	BM25	0.0000	0.4644	0.3958	1.0000
DFR	0.0000	0.3222	0.3042	0.7751	DFR	0.0000	0.4906	0.4722	1.0000
LMJ	0.0000	0.3412	0.3247	0.7751	LMJ	0.0000	0.5074	0.4798	1.0000
LMD	0.0000	0.2961	0.3227	0.7751	LMD	0.0000	0.5163	0.4913	1.0000

Table 4. Evaluation: queries: noun phrases.

Possible Candidates: All sentences in cited paper					Possible Candidates: Method sentences in cited paper				
Precision@k					***Precision@k***				
	Min.	**Median**	**Mean**	**Max.**		**Min.**	**Median**	**Mean**	**Max.**
IBS	0.0000	0.3542	0.3557	1.0000	IBS	0.0000	0.4722	0.4451	1.0000
VSM	0.0000	0.3333	0.3476	1.0000	VSM	0.0000	0.5000	0.4799	1.0000
BM25	0.0000	0.1905	0.2362	0.6667	BM25	0.0000	0.4446	0.3962	1.0000
DFR	0.0000	0.3333	0.3386	1.0000	DFR	0.0000	0.4365	0.4540	1.0000
LMJ	0.0000	0.3333	0.3518	1.0000	LMJ	0.0000	0.4643	0.4704	1.0000
LMD	0.0000	0.3095	0.3114	0.6842	LMD	0.0000	0.5000	0.4988	1.0000
NDCG@k					***NDCG@k***				
	Min.	**Median**	**Mean**	**Max.**		**Min.**	**Median**	**Mean**	**Max.**
IBS	0.0000	0.3120	0.3375	1.0000	IBS	0.0000	0.4464	0.4207	1.0000
VSM	0.0000	0.3040	0.3376	1.0000	VSM	0.0000	0.4748	0.4514	1.0000
BM25	0.0000	0.2224	0.2411	0.7654	BM25	0.0000	0.3986	0.3702	1.0000
DFR	0.0000	0.3024	0.3218	1.0000	DFR	0.0000	0.4450	0.4321	1.0000
LMJ	0.0000	0.3206	0.3590	1.0000	LMJ	0.0000	0.4596	0.4371	1.0000
LMD	0.0000	0.2662	0.2932	0.7654	LMD	0.0000	0.4862	0.4886	1.0000

But this doesn't generalize to every paper as demonstrated in Table 5 which shows, as an example, the per paper results for the four experiment types for the Language Model with Jelinek-Mercer smoothing. The overall statistics of the ranked candidate sentences in the top k are presented in Table 6.

We can notice that fewer 5s and 4s have been ranked as a 0 compared to 3s, 2s and 1s. We think that the 1s, 2s, and 3s may require biochemical knowledge and more background information for the linkage to be effective.

Table 5. Evaluation per paper using language model with Jelinek-Mercer smoothing.

Paper #	Numb. of sents.	Numb. of Method sents.	k	Sentences as Candidates				Method Sentences as Candidates			
				Citation as query		NP as query		Citation as query		NP as query	
				Precision @k	NDCG @k	Precision @k	NDCG @k	Precision @k	NDCG @k	Prec. @k	NDCG @k
1	126	59	19	0.5263	0.5027	0.6842	0.7135	0.6316	0.6741	0.6842	0.7339
2	166	65	16	0.375	0.3547	0.3125	0.2716	0.6875	0.5847	0.625	0.5004
3	150	59	12	0.5833	0.4874	0.4167	0.3792	0.6667	0.5676	0.6667	0.5676
4	162	24	3	0.6667	0.7654	1	1	1	1	1	1
5	194	60	18	0.0556	0.0506	0.2222	0.2828	0.5556	0.576	0.5	0.5344
6	185	44	3	0	0	0.3333	0.2346	0	0	0.3333	0.2346
7	169	53	4	0.5	0.4144	0.25	0.1952	0.5	0.6367	0.25	0.1952
8	291	92	7	0.4286	0.2042	0.4286	0.4791	0.5714	0.4791	0.5714	0.4791
9	233	97	3	0.3333	0.2961	0.3333	0.2961	0.3333	0.2961	0.3333	0.2961
10	224	65	8	0.625	0.7031	0.625	0.7134	0.625	0.4785	0.625	0.4974
11	315	93	31	0.2581	0.3525	0.2258	0.2911	0.4516	0.5278	0.3548	0.4487
12	89	32	5	0.4	0.3452	0.4	0.3452	0.4	0.4704	0.4	0.4704
13	239	47	2	0	0	0	0	0	0	0	0
14	236	103	12	0.25	0.2361	0.25	0.2149	0.4167	0.3857	0.3333	0.2618
15	249	40	14	0.1429	0.1132	0.1429	0.1116	0.4286	0.379	0.4286	0.3774
16	189	99	8	0.25	0.3373	0.375	0.4171	0.5	0.5869	0.625	0.5591
17	112	53	15	0.4	0.4839	0.6	0.6498	0.5333	0.5908	0.7333	0.7721
18	143	65	6	0.6667	0.7751	0.5	0.4593	0.6667	0.7619	0.5	0.429
19	185	62	4	0	0	0	0	0.25	0.2463	0	0
20	165	39	3	0.3333	0.2346	0.3333	0.4693	0.6667	0.5307	0.6667	0.5307
21	266	119	3	0	0	0	0	0.3333	0.2961	0.3333	0.2961
22	170	83	13	0.4615	0.4862	0.3077	0.3733	0.4615	0.4869	0.3846	0.4321

Table 6. Statistics of the ranked candidate sentences over all the papers.

Score	Number Ranked	Number Expected	Percent
5	20	33	(60 %)
4	21	48	(44 %)
3	18	42	(43 %)
2	6	42	(14 %)
1	8	44	(18 %)

60% and 44% of the 5s and 4s are ranked respectively in the top positions. Paper 13 did not yield a candidate sentence for *Precision*@k and *NDCG*@k. This can be due to the following reasons:

1. The number of relevant sentences is very low: 2, whereas the average number of candidate sentences for a paper is 8.
2. The choice of most candidate sentences may involve the use of some inferential information that is not easily translated into the retrieval models at this stage of the study.
3. It is difficult to find the best match for some citations when the matching operation involves external domain specific resources that are not yet available.

Despite these shortcomings, our results show that it is possible to find the set of sentences a citation refers to in a cited paper with reasonable performance.

4.1 Comparison with Other Work

Compared to previous attempts to devise a framework for matching citation text and cited spans in research literature, this current work is novel in certain aspects. The dataset used in [6, 5] is divided into twenty topics, each of which comprises a reference paper and a set of related citing papers. In our work, the citation linkage objective is the same, but we looked at a single citing sentence per paper. Also, we focussed on one particular type of citation, the Method rhetorical category.

This allowed an assessment of a candidate sentence reduction method based on this rhetorical label. The linkage task in this work is a sentence-level matching operation which is intended to avoid the reference text span boundary problem noted in [6] and [5], which is a non-trivial task on its own. And we have assessed the linkage relation on a graded basis rather than a binary classification.

The authors of [6] and [5] reported promising results with query reduction to noun phrases and UMLS expansion. We also have found some, but not general, improvement with the reduction of queries to noun phrases, but we abandoned (possibly prematurely) including associated terminology from thesauri-like sources such as UMLS. We also note our machine learning view of the citation linkage task [15].

5 Conclusion

The citation linkage task aims at finding those sentences in a cited article that are the focus of a citation in a citing paper. We have shown that one way to achieve this goal is to treat the problem as an information retrieval task. While most retrieval techniques usually apply to a large collection of documents, they all involve text matching based on document content similarity. The same matching operation can be achieved at the sentence level as is in the case of the linkage between a citation sentence and its cited sentences in a cited article.

Our results show that the information need that the linkage task tends to achieve can be realized using search engine-like retrieval techniques. We notice that most of the retrieval algorithms provide some good results for the majority of the linkage tasks. However, we need to further investigate why some papers don't yield any relevant sentences. Our hypothesis is that these papers might require domain specific information for the linkage to be achieved. We intend to investigate how to include these resources in the ranking models in future studies.

References

1. Agarwal, S., Yu, H.: Automatically classifying sentences in full-text biomedical articles into introduction, methods, results and discussion. Bioinformatics, vol. 25, no. 23, pp. 3174–3180 (2009) doi: 10.1093/bioinformatics/btp548

2. Agirre, E., Diab, M., Cer, D., Gonzalez-Agirre, A.: SemEval-2012 task 6: A pilot on semantic textual similarity. In: Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, pp. 385–393 (2012)
3. Amati, G., Van Rijsbergen, C. J.: Probabilistic models of information retrieval based on measuring the divergence from randomness. ACM Transactions on Information Systems, vol. 20, no. 4, pp. 357–389 (2002) doi: 10.1145/582415.582416
4. Białecki, A., Muir, R., Ingersoll, G.: Apache Lucene 4. In: Proceedings of the SIGIR 2012 Workshop on Open Source Information Retrieval, pp. 17–24 (2012)
5. Cohan, A., Soldaini, L., Goharian, N.: Matching citation text and cited spans in biomedical literature: A search-oriented approach. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1042–1048 (2015) doi: 10.3115/v1/N15-1110
6. Cohan, A., Soldaini, L., Mengle, S. S., Goharian, N.: Towards citation-based summarization of biomedical literature. In: Proceedings of the Text Analysis Conference (2014)
7. Dagan, I., Glickman, O., Magnini, B.: The PASCAL recognising textual entailment challenge. In: Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment, vol. 3944, pp. 177–190 (2006) doi: 10.1007/11736790_9
8. Dolan, B., Quirk, C., Brockett, C.: Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In: Proceedings of the 20th International Conference on Computational Linguistics, pp. 350–356 (2004) doi: 10.3115/1220355.1220406
9. Garfield, E.: Science citation index–A new dimension in indexing: This unique approach underlies versatile bibliographic systems for communicating and evaluating information. Science, vol. 144, no. 3619, pp. 649–654 (1964) doi: 10.1126/science.144.3619.649
10. Garfield, E.: Citation analysis as a tool in journal evaluation: Journals can be ranked by frequency and impact of citations for science policy studies. Science, vol. 178, no. 4060, pp. 471–479 (1972) doi: 10.1126/science.178.4060.471
11. Garfield, E.: Is citation analysis a legitimate evaluation tool? Scientometrics, vol. 1, no. 4, pp. 359–375 (1979) doi: 10.1007/BF02019306
12. Garzone, M., Mercer, R. E.: Towards an automated citation classifier. In: Conference of the Canadian Society for Computational Studies of Intelligence, vol. 1822, pp. 337–346 (2000) doi: 10.1007/3-540-45486-1_28
13. Harter, S. P.: A probabilistic approach to automatic keyword indexing. Part II. An algorithm for probabilistic indexing. Journal of the American Society for Information Science, vol. 26, no. 5, pp. 280–289 (1975) doi: 10.1002/asi.4630260504
14. Houngbo, H., Mercer, R. E.: An automated method to build a corpus of rhetorically-classified sentences in biomedical texts. In: Proceedings of the First Workshop on Argumentation Mining, pp. 19–23 (2014) doi: 10.3115/v1/W14-2103
15. Houngbo, H., Mercer, R. E.: Investigating citation linkage with machine learning. In: Proceedings of the 30th Canadian Conference on Artificial Intelligence, pp. 78–83 (2017) doi: 10.1007/978-3-319-57351-9_10
16. Houngbo, H. K.: Investigating citation linkage between research articles. Ph.D. thesis, Electronic Thesis and Dissertation Repository, The University of Western Ontario (2017)
17. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Transactions on Information Systems, vol. 20, no. 4, pp. 422–446 (2002) doi: 10.1145/582415.582418
18. Mercer, R.: Locating and extracting key components of argumentation from scholarly scientific writing. vol. 6, no. 4, pp. 3–15 (2016)

19. Nakov, P., Schwartz, A., Hearst, M. A.: Citances: Citation sentences for semantic analysis of bioscience text. In: Proceedings of the SIGIR'04 Workshop on Search and Discovery in Bioinformatics (2004)
20. Palau, R. M., Moens, M. F.: Argumentation mining: The detection, classification and structure of arguments in text. In: Proceedings of the 12th International Conference on Artificial Intelligence and Law, pp. 98–107 (2009) doi: 10.1145/1568234.1568246
21. Radev, D. R., Jing, H., Budzikowska, M.: Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In: Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization, vol. 4, pp. 21–30 (2000) doi: 10.3115/1117575.1117578
22. Robertson, S., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond. vol. 3, no. 4, pp. 333–389 (2009) doi: 10.1561/1500000019
23. Small, H.: Cited documents as concept symbols. Social Studies of Science, vol. 8, no. 3, pp. 327–340 (1978) doi: 10.1177/0306312778008003
24. Teufel, S., Siddharthan, A., Tidhar, D.: Automatic classification of citation function. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, pp. 103–110 (2006)
25. Zhai, C.: Statistical language models for information retrieval. Synthesis Lectures on Human Language Technologies (2009) doi: 10.1007/978-3-031-02130-5
26. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to Ad Hoc information retrieval. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 334–342 (2001) doi: 10.1145/383952.384019

ISSN 1870-4069

Semantic Integration of COPEs in GOLD Ontology

Malek Lhioui[1], Kais Haddar[1], Laurent Romary[2]

[1] Sfax University,
Laboratory MIRACL, Multimedia, Information Systems and Advanced Computing Laboratory,
Tunisia

[2] Inria & Centre Marc Bloch,
Germany

laurent.romary@inria.fr, ma.lhioui@gmail.com,
kais.haddar@yahoo.fr

Abstract. This paper has as goal the semantic integration of the local ontologies named COPEs (Community of Practice Extension) in GOLD (General Ontology for Linguistic Description) ontology. COPEs are OWL ontologies that include specific knowledge. However, GOLD is a global ontology that includes general knowledge of the field. Thus, we deal with the challenge of the construction of a global schema presented as global ontology for lexical resources introduced as local ontologies. The originality of our suggestion consists on the use of a web semantic method to resolve an NLP issue, i.e. semantic integration as method for interoperability resolution between lexical resources. We define lexicons as local ontologies using Description Logics. Then, we build the resulted global ontology by combining alignment techniques and logical-reasoning.

Keywords: Semantic integration, COPEs, GOLD, interoperability.

1 Introduction

As the need for interoperability between lexical resources is increasing, we deal with the challenge of building a global schema for lexical resources. Indeed, several linguistic consortiums, such as ISO, need to share particular fragments of specific domains. Users will not be obliged to learn about the details of many sources like structures, vocabularies, concepts, relations, etc. Consequently, users are conciliated when formulating queries. There are several formalisms of lexical resources such LMF, TEI, HPSG, etc. However, formalisms may dispose of incomplete and partial data because some of them do not offer full knowledge. For example, TEI does not offer complete semantic information.

For this purpose, linguists need to exchange information between lexical resources. The exchange of information seems to be necessary to provide a more complete linguistic resource which reply to users' needs.

adfa, p. 1, 2011.
pp. 71–81; rec. 2017-02-08; acc. 2017-04-12

The method presented here allows constructing a global schema: ontology treated as pivot format for lexical resources. For this construction, we use semantic integration because of its large interest dealing with heterogeneous knowledge, ensuring enrichment and avoiding destruction of old editions.

Building such a global ontology for reasons of interoperability between lexical resources may face several problems. First, we have to perfectly choose the optimal representation language for our case (RDF(S), OWL-Lite and OWL-DL). Besides, the construction of such ontology requires a precise knowledge of the lexical resources, their heterogeneity in the distribution of knowledge, the used nomenclature and their coverages (lexical, syntactic, etc.). From a technical point of view, the collection of lexical resources is a big dilemma for the majority of specialists in the language community. In our case, we use ALIF platform so we need to learn carefully about it.

Our method is original since there are no previous operable works trying to resolve interoperability between lexical resources. Moreover, using ontologies for resolving a big issue which is interoperability between lexical resources is in itself an innovation. From another point of view, interoperability nowadays become a big issue and recent projects must take care of it otherwise they are out of progress. The [1] report states that: “The lack of interoperability costs the translation industry a fortune”. Fortune is compensated to regulate the adjustment of lexical resources.

The method we already introduce in this paper is operable whatever the language. In the following parts of the paper, we give a concise state of the art talking about our big topic which is interoperability between lexical resources. Then, we define prerequisites required in introducing our method. Then, we define our proper new method which is using semantic integration between local ontologies to build global one. Finally, we bring to a close with experimentation and evaluation section.

2 Related Works

There are no previous works dealing with the use of semantic integration for the purpose of resolving interoperability between lexical resources. However, there are works related with semantic integration and others concerning interoperability between lexical resources. Since there are two main separated topics, we classify the following state of the art into two main parts. The first part concerns the semantic integration. In the second part, we discuss interoperability between existing lexical resources.

2.1 Semantic Integration

Semantic integration is a recent approach which is based on ontology integration. In order to formalize this approach, experts use Description Logics and ontology alignment. In [2], authors present a distributed description logic.

The formalism presented in [3] defines aspects of distributed and modular ontology reasoning. In the two cases, authors try to define that the concept of distributed description logics is needed for relating various data sources. In [4], authors introduce a new approach “E-connections framework” as a solution for connecting different sources. The defined approach is.

2.2 Interoperability between Lexical Resources

Since there are no serious cited efforts in literature aiming to resolve interoperability between lexical resources in NLP domain, we discuss the bidirectional mapping between formats of lexical resources. [6] is the first mapping process converting HPSG lexicons to OWL ontology. A rule-based system is invented by [7] in order to translate LMF syntactic lexicon into TDL using the LKB platform. Then, a prototype for projection HPSG syntactic lexica towards LMF have been developed by [8]. In the same context, a mapping process converting LMF lexicons to OWL ontologies is described in [9].

These works usually involve two formalisms, processing more than two formats is a hard task even impossible. In order to appease the difficulty of transformation process, the ISO try to solve the problem with a normalization process. It proposes an ISO standard in 2003named LMF [10]. All these works are deeply linked to our proposed method. However, we use the approach of semantic integration to introduce a new method for resolving interoperability between lexical resources. In the following part, we familiarize with notions required to define our method.

3 Prerequisites

We use for semantic integration reference ontology: on the one hand links between source ontologies are obtained from the taxonomical relationships of the reference ontology. On the other hand, mappings between the global ontology and sources are obtained by syntactic-matching, from source-concepts names to reference-ontology-concepts names.

3.1 Ontology of Reference: GOLD

GOLD[1] (General Ontology for Linguistic Description) is a general ontology described in OWL including linguistic knowledge as well as a qualified linguist [12]. Knowledge including in this ontology consists on the core of any theoretical framework. Furthermore, GOLD incorporates knowledge concerning descriptive linguistics. For example, “an adjective is a part of speech” [12]. Linguistics communities consider GOLD as a reference ontology that uses language-neutral and theory-neutral terminology. For example, LexicalResource is a subclass of gold: Entity.

3.2 Local Ontologies: COPEs

Local ontologies are sub-communities of practice considered as instantiations noted COPEs (Communities of Practice Extension (COPEs). COPEs are simple OWL ontologies that import local knowledge to a global resource [11]. In order to give a real example of components in COPE of LMF, we note the part of speech propriety designed as subclasses of lmf: pos. Parts of speech in LMF have perhaps their

[1] http://www.linguistics-ontology.org

equivalent in GOLD ontology. The liaison will be done automatically by means of semantic integration. This phenomenon is described clearly in the next section.

3.3 Semantic Integration

Data structures whatever their kinds (non-structured, semi-structured and structured) are more and more complex. Therefore, their handling is no longer simple. Indeed, data present an accessibility issue because of its different kinds. However, though their heterogeneity, several data sources are semantically related. Different concepts describe the same reality. The access to these data constitutes a big dilemma because of the non-precision of their localization. Consequently, interoperability is so required in this case between a new created system playing the role of interface and the other sources. The new created system is based on a data integration process offering a new interface for distributed, heterogeneous and independent sources.

4 Semantic Integration of COPEs in GOLD Ontology

Data semantic integration is in general progress with the evolution of data structures (XML, RDF, OWL, etc.). The goal of our proposed paper is to build global ontology from local ontologies using of reference ontology GOLD using semantic integration. In order to formalize our domain, we exploit description logics DL to define a domain by a set of:

– Concepts: which express classes and manipulate them as objects, example: Lexical Entry, Lemma, Stem, etc.
– Roles: which express relations and operate them as relations between objects, example: Lexical Entry related To Lemma.

An ontology $O = < T, A >$ is composed of:

– $TBox\ T(intensional\ Knowledge)$: Identify general propriety of concepts and roles,

In order to illustrate ontology components, we give the following example:

$\exists\ relatedTo\ LexicalEntry$

$\exists\ relatedTo\ Lemma$

$\exists\ hasForm\ LemmatisedForm$

$\exists\ hasForm$ InflectedForm

$InflectedForm \subseteq LemmatisedForm$

$InflectedForm \subseteq \delta(writtenForm)$

$InflectedForm \subseteq \delta(number)$

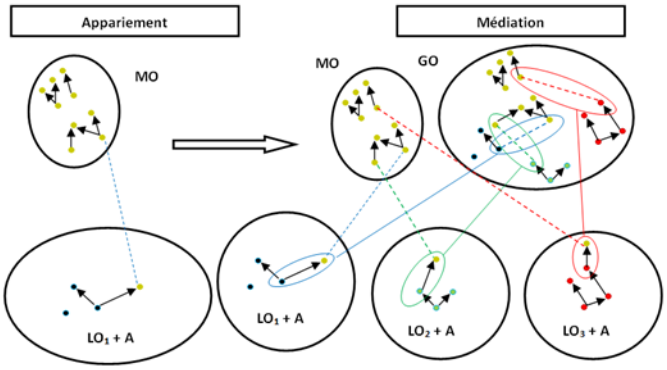
$\rho(writtenForm) \subseteq \delta(xsd{:}\,string)$

$\rho(number) \subseteq \delta(xsd{:}\,string)$

– $ABox\ A\ (extensional\ knowledge)$: Identify assertions related to concepts and roles instances.

In order to more explain the composition of the ontology, we give a real example:

– $InflectedForm(inflectedForm)$
$writtenForm(inflectedForm, clergymen)$
$number(inflectedForm, plural)$



MO: Mediator ontology
GO: Global ontology
LO_i: Local ontology
A: Anchoring

Fig. 1. Proposed method.

After presenting DL-ontologies, we define the required foundations in our proposal. Here are three mains needed $TBox$es in our proposed method:

– A set of local $TBox$ called$\{Tli\}$: They present involved local data sources {Si} in the sharing process.

– A $TBox\ Tm$: It provides intentional knowledge extracted from the ontology of reference.

In order to more explain the composition of the ontology, we give a real example:

– A Tg : conciliates the different local $TBox$ and supplies a shared conceptual level of the domain application.

After this formalization, the issue of construction a global ontology is summarized in the build of $TBox\ Tg$. This Tg integrates $TBox$ of local ontologies and adjusts their concepts using $TBox$ of the reference mediator.

The previous fig.1 shows the two main steps of the proposed method: appariement and mediation. The method is based on the use of automatic reasoning functions of description logics in order to automate the construction process. Then, we use ontology of reference to have an appropriate conceptualization of the application domain. The ontology of reference has been developed independently from any specific objective by experts in knowledge and domain engineering: GOLD.

4.1 Appariement

The appariement step is based on modifications made in $Tl\ TBoxes$. Then, we integrate them in global $Tg\ TBox$. Fig. 2 shows the appariement step. Two sub steps have been to achieve: *anchoring process* and *automatic updating of local* $Tl\ TBox$ into Tla. This step allows linking concepts of Tl (anchored concepts) and concepts of referene mediator $TBox\ Tm$ (anchor concepts).

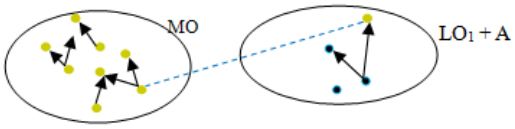


Fig. 2. Appariement of concepts between the mediator and the local ontology.

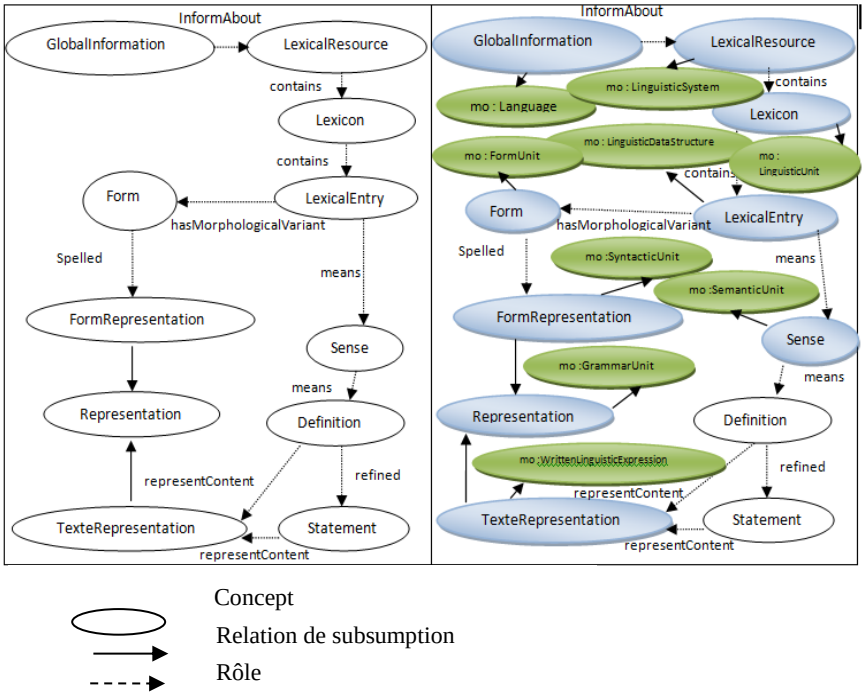


Fig. 3. Concrete example of the appariement step: application in LMF core model.

Fig. 2 summarizes the whole process of appariement step: anchoring and updating. In the anchoring process, we make liaison between concepts of the mediator (MO) and those of local ontology (LO). Fig. 3 shows a real example of the appariement step.

In fig. 3, two types of anchoring are established: lexical and semantic anchoring. For the first anchoring: It is simply matchings of Tl to Tm :

– Calculation of a set of mappings noted $M = \{mi\} \ / \ mi = Al \sqsubseteq Am \ \ (Al \in Tl, Am \in Tm)$.

– Using of lexical similarity measure δ: $\lceil NL \times \lceil Nm \rightarrow [0, 1]$ ($\lceil NL, \lceil Nm$ are a set of atomic concepts names.

Semantic anchoring: It is a question of finding additional anchoring concepts which are subsumed by anchored process.

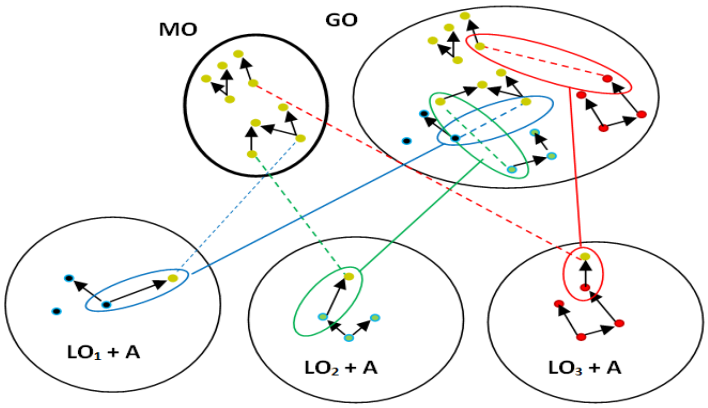


Fig. 4. Global procedure of the mediation.

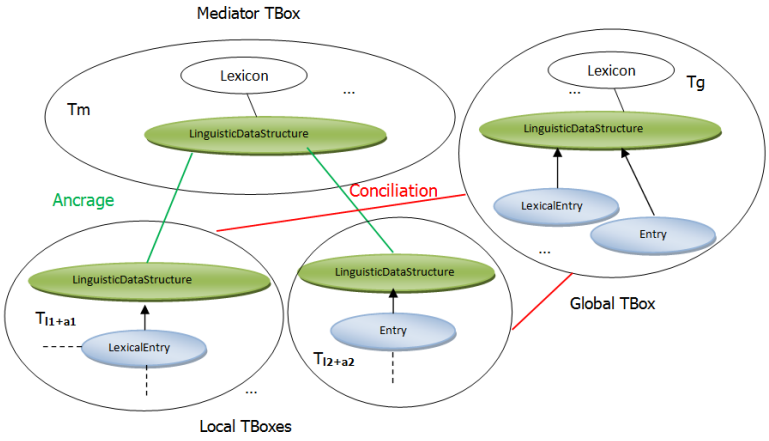


Fig. 5. Real example of mediation process.

After the anchoring process, we build a new Tla containing the result of the appariement:

We note $Tla =< Tl, M >$ the appariement of Tl compared to Tm.

– Tl is a local ontology.

– M is the result of anchoring Tl compared to Tm.

4.2 Mediation

The mediation consists on the integration process based on *related concepts computation for* Tla. The mediation step allows the construction of the global TBox Tg using the result of the appariement phase. The new ontology is reached *incrementally*

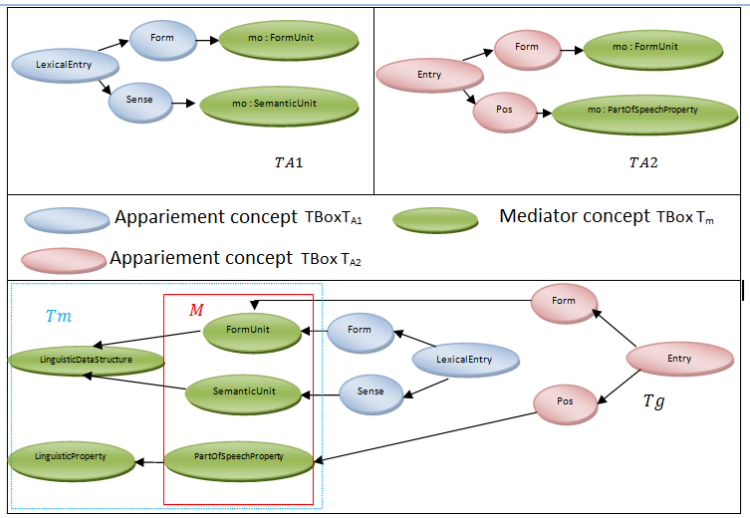


Fig. 6. Illustration of mediation algorithm.

by integrating the resulted TBoxes of the appariement step. It is a question of linking appariement TBoxes concepts. Fig. 4 illustrates the whole process of the mediation.

In fig. 4, the global ontology GO is incrementally built in two main sub-steps: *integrate local ontology and their appariement Tla* and *calculate all related subsumers for the anchor concepts*.

The first sub-step is summarized in the liaison between the resulted TBoxes from the appariement step and those of the mediator ontology. Concepts in the global ontology GO are linked incrementally with anchor concepts in the mediator ontology MO. Thus, the TBox Tg includes the following sets: - the set of appariement TBoxes *Tla*, and – a subset of Tm containing the related part of the hierarchy related with anchored concepts.

In order to give a real example between LMF and TEI ontology, we take the example in fig. 5. In this example the concepts *LexicalEntry* of the appariement TBox Tl_1+a_1 and *Entry* of the appariement TBox Tl_2+a_2 are respectively anchored by the same concept of the mediator TBox Tm: *LinguisticDataStructure*.

The structure of the mediator TBox reveals that *LinguisticDataStructure* has related concept. We add this relation to merge the concepts LexicalEntry and Entry in the global TBox Tg.

The fig. 5 can be generalized by the following algorithm of mediation:

- Input: $\{T_{Ai} = < T_{li+ai}, Mi >\}$, Tm
- Output: $Tg = < \{T_{Ai}\}, Tm >$

Each anchor concept Am of Mi is integrated in the Tg hierarchy.

Calculation of related concepts of Am in Tm called Arc (All Related Concepts) among present concepts in the hierarchy Tm

$Arc \leftarrow All\_related_{Tm}\ (Am)$

The previous algorithm is illustrated as follow:

$$T_{A1} = < T_{l1+a1}, M1 >$$

$$M1 = \{\text{Form} \subseteq \text{mo : FormUnit, Sense} \subseteq \text{mo : SemanticUnit}\}$$

$$T_{A2} = < T_{l2+a2}, M2 >$$

$$M2 = \{\text{Entry} \subseteq \text{mo : LinguisticDataStructure, Form} \subseteq \text{mo : Form}\}$$

In fig.6, we illustrate the previous algorithm of mediation using reference ontology GOLD as mediator (concepts colored with green) and logical-inference of description logics for automatic construction of the global ontology.

5 Experimentations and Evaluations

In order to evaluate our approach, a system of request is established using a construction process of requests exploiting special types of mappings. The build system is able to ask the appropriate local ontologies. The system interrogates the concerned sources and recomposes partial responses and restitutes the global response. This system is set up in a separated module but not explained in this paper.

After the description of the approach aiming to build automatically a global ontology, we introduce the different experimentations done in this section. In order to verify that our approach is feasible, we have to experiment it in order to prove that the approach allows producing a global ontology providing a shared conceptualization for involved sources with reasonable costs in terms of time and human resources. Our approach provides a facility of updating sources.

In order to evaluate our proposal, we choose a full ontology of reference in the lexical resources' domain. We realize the experimentations with the following techniques characteristics: a machine endows with a system Windows 7 with an Intel(R) Core (TM) processor. For the appariement step, we have used a well-known measure of lexical similarity called "string metrics" proposed by [13]. All local used ontologies are built automatically using an MDA[2] process [14]. The following notations used in the experimentation part are:

- LM: Lexical Mappings found after a lexical appariement, additional notations are given such 1;1 and 1;m used respectively to design mappings for one appariement found for one concept and mappings found for several candidates found for the same concept,
- VM: Validate Mappings,
- SM: Mappings found after a semantic appariement.

The table 1.1 illustrates results obtained after the appariement step for the two ontologies: LMF core ontology and TEI ontology. The two cited ontology are built automatically in [14] via the MDA approach.

[2] MDA : Model Driven Architecture

Table 1. Results of the appariement process in the context of LMF and TEI ontologies.

Ontologies	Concepts	Appariement							
		LM		VM	SM	Concepts selected for appariement			Time for execution
		1;1	1;m			pairings	selected	matching	
TEI Ontology	207	203	1	197	9	206	1	178	02 :39s
LMF Ontology	39	37	1	34	5	39	0	17	00 :49s

Table 2. Results of the mediation process in the context of LMF and TEI ontologies.

Pairings	Concepts for matchings	Mediation		
		Global ontology		Mediation time
		Local concepts	Concepts given by the reference ontology	
Ta1	178	207	83	05 :03s
Ta2	17	175	104	02 :01s

When analyzing results shown by the given table, we note that almost lexical mappings are 1;1. Among these found mappings, only 197 ones are valid. We note that many concepts selected for appariement share the same pairing: we have 206 pairings with 178 matchings. We remark that the time for execution reflects complexity of the construction process.

According to the results given by the table 2, the mediation time is not proportional to the concepts given because it depends on the process integration and not the number of concepts found for mediation. In fact, the integration is the task that consumes the time. Thus, the complexity is linear to mediation process.

6 Conclusion

In this proposal, we have proposed a new method for interoperability between lexical resources using semantic integration of lexical local ontologies in GOLD. Our method does not depend on the quality of involved lexical resources. The established method combines the results of MDA approach and ontology alignment techniques to make semantic links between involved lexical resources. Our research field contains four main parts: lexical resources, interoperability issue, semantic integration and ontologies alignment.

First of all, we have made a smart research on the main existing lexical resources in several languages. Then, we have made a great study on interoperability issue, and since there are no serious attempts to resolve this notion in NLP area, we have discussed the bidirectional mapping from one format to another. In future works, we have to extend our method using other metrics of the two-alignment ontology. In fact, interoperability appears to be solved since it combines two main approaches MDA Transformation and ontology alignment.

References

1. TAUS: Report on a TAUS research about translation interoperability (2011)
2. Borgida, A., Serafini, L.: Distributed description logics: Assimilating information from peer sources. Journal on Data Semantics I, Springer, vol. 2800, pp. 153–184 (2003)
3. Serafini, L., Borgida, A., Tamilin, A.: Aspects of distributed and modular ontology reasoning. IJCAI, pp. 570–575 (2005)
4. Grau, B. C., Parsia, B., Sirin, E.: Combining OWL ontologies using E-connections. Journal Web Semantics, vol. 4, no. 1, pp. 40–59 (2006) doi: 10.1013/j.websem.2005.09.010
5. Niang, C., Bouchou, B., Lo, M., Sam, Y.: Automatic building of an appropriate global ontology. In: East European Conference on Advances in Databases and Information Systems, pp. 429–443 (2011)
6. Wilcock, G.: An OWL ontology for HPSG. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions, pp 169–172 (2007)
7. Loukil, N., Ktari, R., Haddar, K., Benhamadou, A.: A normalized syntactic lexicon for arabic verbs and its evaluation within the LKB platform. ACSE (2010)
8. Haddar, K., Fehri, H., Romary, L.: A prototype for projecting HPSG syntactic lexica towards LMF, JLCL (2012)
9. Lhioui, M., Haddar, K., Romary, L: A prototype for projecting LMF lexica towards OWL, CICLING (2015)
10. Francopoulo, G.: LMF lexical markup framework. John Wiley & Sons, Inc (2013)
11. Farrar, S., Lewis, W. D.: The gold community of practice: An infrastructure for linguistic data on the web (2005) http://www.u.arizona.edu/˜farrar/
12. Stoilos, G., Stamou, G., Kollias, S.: A string metric for ontology alignment. In: Proceedings of the 4rd International Semantic Web Conference (ISWC), Lecture Notes in Computer Science, Springer, vol. 3729, pp. 624–637 (2005)
13. Lhioui, M., Haddar, K., Romary, L.: A new method for interoperability between lexical resources using MDA approach. Advanced Intelligent Systems and Informatics (2016)

ISSN 1870-4069

Creating Parallel Arabic Dialect Corpus: Pitfalls to Avoid

Salima Harrat[1], Karima Meftouh[2], Kamel Smaili[3]

[1] Ecole Nationale Supérieure d'Informatique,
Algeria

[2] Badji Mokhtar University,
Algeria

[3] Campus Scientifique LORIA,
France

{erika.siebert-cole, peter.strasser, lncs}@springer.com

Abstract. Creating parallel corpora is a difficult issue that many researches try to deal with. In the context of under-resourced languages like Arabic dialects, this issue is more complicated due to the nature of these spoken languages. In this paper, we share our experiment of creating a Parallel Corpus which contain several dialects all aligned with Modern Standard Arabic (MSA). We attempt to highlight the most important choices that we did and how good were these choices.

Keywords: Modern standard Arabic, Arabic dialects, dialectal corpora, parallel corpora.

1 Introduction

In overall Arab world, Modern Standard Arabic (MSA) is the official language. It is used in the formal speeches, newspapers, official documents, press releases, education etc. In parallel of this, in everyday life, Arab people use dialects. They are spoken languages highly influenced by MSA but also by other foreign languages like French, Spanish, English and Turkish. These languages, have emerged on forums, social media and TV shows; and even in the last years, an important number of movies and series are subtitled in these languages whereas before they were translated to MSA.

The diglossia phenomenon[1] is more accentuated on the internet and mobile telephony, and has created new needs in the area of NLP related to the Arabic language. In fact, all NLP applications dedicated to the Arabic language like machine translation, speech recognition, speech synthesis, sentiment analysis... do not take into account the Arabic dialects. These dialects could be considered as under-resourced languages as they lack NLP resources.

[1] A situation in which two distinct varieties of a language are spoken within the same speech community

In the same vein, dealing with Arabic dialects is more complicated than MSA in NLP area. Indeed, MSA is a natural language with a full writing system including strict rules that cover all orthographic varieties, a rich morphology and a strong syntactic system. In contrast, Arabic dialects are spoken languages with no writing system.

They differ from one Arab country to another and they simplify most Arabic language rules. Creating resources for Arabic dialects presents additional challenges to Arabic NLP. If MSA monolingual corpora, for instance, are available with very acceptable sizes and many parallel corpora including Arabic are downloadable for free, most of the Arabic dialects are still under-resourced in terms of such data. Monolingual and parallel textual corpora for some dialects exist but they are few in number which is due to the fact that they are time-consuming.

In this paper, we describe an experiment of creating a dialectal parallel corpus that includes several Arabic dialects in addition to MSA. This corpus has been created for the purpose of machine translation. We try to evaluate our experiment and highlight the most important conclusions that we made. The rest of the paper is organized as follows: in Section 2, we give an overview of Arabic language and its dialects whereas Section 3 presents the most important works dedicated to creating textual resources for Arabic dialect. Section 4 describes the parallel dialectal corpus that we created and in Section 5, we mention the most important choices that we made and how good they were. Section 6 concludes this work.

2 Arabic Language and its Dialects

Arabic is a generic term that covers: Classical Arabic, Modern Standard Arabic and Arabic dialects. The first term is related to the Arabic used in the Qur'an and in the earliest literature from the Arabian peninsula, and it is still used in the literature. Modern Standard Arabic is a modern variant of classical Arabic. In all Arab countries, it is taught in school and used in written correspondence, religious discourses,... but not in everyday conversations nor in households.

Arabic dialects or colloquial Arabic are spoken varieties of the Arabic language. They are influenced by European languages such as French, Spanish, English, and Italian because most of the Arab countries were European colonies during the 19th century. Arabic dialects are described in the literature according to east-west dichotomy [11]; Maghreb dialects regroup dialects spoken in north Africa: Algeria, Tunisia, Morocco, Libya and Mauritania while Middle-east dialects regroups the dialects spoken in Middle-east countries: Levantine dialect (Syria, Lebanese, Palestinian and Jordan), dialect of Arabian peninsula (Gulf countries and Yemen), Iraqi dialect Egyptian and Sudan dialect.

3 Textual Resources for Arabic Dialects

Availability of resources is a big issue for NLP applications dedicated to Arabic dialects. In [19], authors drew a survey of available Arabic corpora and lexicons and underlined the lack of this kind of resources for Arabic dialects. Several works are dedicated to build such resources.

Authors of [9] elaborated a Levantine lexicon using transductive learning. For Iraqi dialects, authors of [10] used an annotated recorded speech to build a lexicon, whereas in [18] a spelling corrector was presented for this dialect. For Tunisian dialect, authors of [21] created a lexicon by converting MSA patterns to Tunisian dialect patterns and then extracting specific roots and patterns from a training corpus that they created for this purpose.

This Tunisian dialect lexicon was then used to adapt Al-Khalil [5], a morphological analyzer. Another work dedicated to Tunisian dialect is described in [6], where authors used the Penn Arabic Treebank (PATB) [12, 15] to create a Tunisian dialect text corpus and a bilingual dictionary (MSA/Tunisian dialect) by using a transformation method based on the parts of speech of ATB words. In [1] a lexicon for MSA/Egyptian dialect (of Cairo the capital city of Egypt) was created. All dialect entries are mapped to their MSA synonyms and thus they get access to the quite available NLP tools and resources for MSA.

Regards to textual corpora, several works attempted to create this kind of resources. Authors of [13] developed a treebank for the Levantine dialect which consisted of morphological and syntactic annotation of approximately 26K words of Levantine Arabic conversational telephone speech. Also, a treebank for Egyptien dialect was built in [14] in addition to a morphological analyzer. In [2] YADAC (a multi-genre Dialectal Arabic (DA) corpus) is created for Egyptian dialect using Web data from microblogs (Twitter), blogs/forums and online knowledge market services.

Other important resources for Arabic dialects are multi-dialects corpora. Several researches were dedicated to creating such kinds of resources like [3] where the web was used as a source to build dialect corpus for Gulf, Levantine, Egyptian and North African dialects, or [8] where authors presented a multi-dialect, multi-genre, human annotated corpus of Levantine, Gulf, Egyptian, Iraqi and Maghrebi dialects. In [17], authors collected also a multi-dialect corpus from Twitter for a large set of Arabic dialects.

In terms of Arabic dialect parallel corpora, resources are more scarce because of the required effort to build them. In [4], a multidialectal Arabic parallel corpus is presented, it is a collection of 2K sentences in MSA, Egyptian, Tunisian, Jordanian, Palestinian and Syrian Arabic, in addition to English. The starting point of this work was the 2K Egyptian sentences extracted from the Egyptian part of the Egyptian-English corpus built in [20]. As described in [16], a parallel dialectal corpus (PADIC) including several dialects and MSA was built from scratch. The rest of this paper is about the creation of this corpus, we made a number of observations and conclusions that we have summarized below.

4 A Brief Overview of PADIC, the Parallel Arabic Dialect Corpus

In this section, we give a brief description of our parallel corpus and the methodology of its creation. The corpus includes in addition to MSA, fives dialects, namely Algerian dialects (from Algiers the capital city of Algeria and Annaba a city in the east of the country), Tunisian, Moroccan, Palestinian and Syrian. For creating the Algiers dialect corpus (ALG), we have transcribed the scenarios of films and sitcoms.

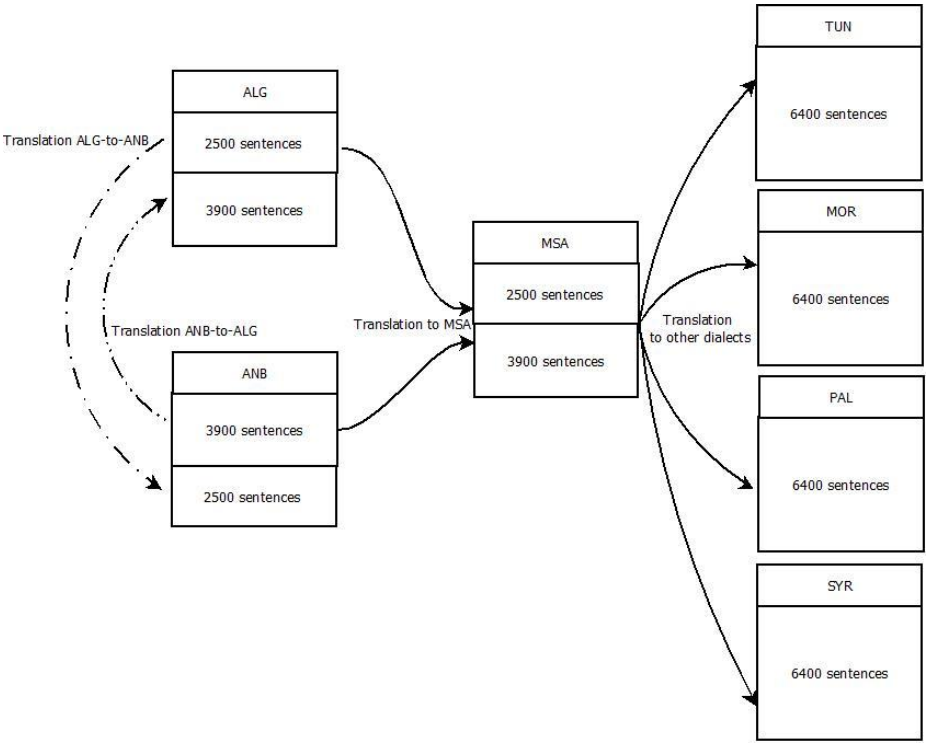


Fig. 1. Parallel arabic dialect corpus creation.

This transcription allowed the collection of 2,5K sentences that we translated into standard Arabic. A corpus of Annaba dialect(ANB) was also created in the same way but by transcribing recordings of the daily life of some people of Annaba (university, doctor's waiting room, households).

This transcription allowed to collect 3,9K sentences which were also translated into standard Arabic. In order to increase the size of the two corpora, each of them has been translated to the other. Thus, at the end of this operation, a first trilingual (MSA, ALG, ANB) corpus of 6,4K sentences was created.

In order to introduce the Tunisian, Moroccan, Syrian and Palestinian dialects, we used MSA as a pivot language. The standard Arabic part of the tri-lingual corpus constructed as explained above has been translated to Tunisian, Moroccan, Syrian and Palestinian (see Figure 1).

The Tunisian corpus was produced by a number of native speakers from the south of Tunisia. The Moroccan, Syrian and Palestinian corpora were created in the same way as the Tunisian, except that each one was produced by only one native speaker of each dialect. The Palestinian dialect concerned by this study is mainly the dialect of Gaza Strip. The Syrian dialect is that of the city of Damascus, whereas the Moroccan is the dialect of Rabat (the capital city of Morocco). Briefly, the parallel corpus consists of seven texts of $6.4K$ sentences in Algerian dialect (two texts), Tunisian, Moroccan, Palestinian, Syrian in addition to MSA.

Table 1. Perplexity values computed for each language model of the parallel corpus.

Language/Dialect	MSA	ALG	ANB	TUN	MOR	SYR	PAL
Perplexity	51.79	57.79	61.49	62.96	58.05	59.88	56.09

The average size of each corpus is about 41477, while the average number of distinct words is 9815. It is important to note that all operations described above were done manually. It was time-consuming. A group of persons (native speakers of the concerned dialects) contributed to PADIC creation: twenty persons for ANB and TUN and one person for the other dialects. It should be noted that these people did this work for free.

In order to have an idea about each monolingual corpus, we computed related perplexity which is frequently used as a quality measure for language models [7]. Each side of the parallel corpus was split into test and training set of 800 and 5600 sentences, respectively. We used tri-grams language models for which we calculated perplexity values as shown in Table 1.

It is clear that the lowest perplexity is related to MSA, this was expected since MSA is a natural language with a strong orthographic system. The highest values are those for ANB and TUN. This could be explained by the number of persons who participated in creating these two corpora (twenty persons for each corpus). Indeed, Every person used its own dialectal words and wrote them with little regard for orthography.

5 Pitfalls to Avoid when Creating Dialectal Corpora

In this section, we discuss the most important points that must be taken into account when creating parallel corpora for Arabic dialects.

5.1 Pivoting Throw Close Languages

Pivoting throw MSA was an intuitive choice since Arabic dialects are variants of this language. Unfortunately, we noticed this pivoting has in some way influenced the translators. Indeed, we observed that they tend to apply the Arabic sentences structure to dialectal ones when translating.

The influence of standard Arabic on dialect corpora is clear in all of them translated from MSA (Tunisian, Moroccan, Palestinian and Syrian). We give in Table 2 some examples collected from the parallel corpus where the influence of MSA is noticeable.

In the same vein, it is not recommended to translate between dialects, especially close ones. In fact, the translator could be easily influenced by the source dialect and unconsciously uses words or terms belonging to the source dialect even if they are rarely used in the target dialect.

We think that pivoting throw a foreign language such as French (for Maghreb dialects) or English (for middle-east dialects) could produce ”more dialectal” sentences because the distance between languages imposes the use of purely dialectal words since borrowing words is not acceptable as the case in translating from MSA.

Table 2. Examples of MSA pivoting influence from the parallel corpus.

Dialect/Language	Sentence	Meaning
MSA	هذا شيء غير معقول	It is impossible
ALG	ماكانش منها هادي	
ANB	ماكانش منها ادي	
TUN	مهوش معقول	
MOR	هاد الشي ماشي معقول	
SYR	هالشي مو معقول بنوب	
PAL	هذا اشي مش معقول	
MSA	و هل في الَامر شك	Is there a doubt
ALG	و هَادي فيهَا هَدرة	
ANB	و ادي فَاهَا هَدرة	
TUN	عندك شك	
MOR	واش كاين شي شك في دكشي	
SYR	مَا في شك	
PAL	و هو الموضوع في شك	
MSA	شخصيَا لَا يمكن ان امر	Personally I can not pass
ALG	انَا مَا نقدرش نجوز	
ANB	انَا مَا نقدرش نعدي	
TUN	شخصيا منجمش نمر	
MOR	شخصيا منقدش نعدي	
SYR	شخصيَا مَا بقدرش اني امر	
PAL	شخصيَا انَا مَا بمر	

5.2 Choosing Topic

Regards to topics, we did not impose any restrictions. Users charged to record conversations were free to record any kind of discussion. We aimed to diversify subjects in order to have as broad a vocabulary as possible. But we noticed that this diversification has generated some conversations in particular domains like religion and studies with specific vocabularies.

For example, Arabic people when speaking about religion (whatever is their religion) tend to use standard Arabic. They switch from dialect to standard Arabic spontaneously, especially when they evoke religious texts. Due to that, some passages in dialectal corpora include standard Arabic expressions that translator left unchanged. We give in Table 3 some Arabic religious expressions present in all dialectal corpora.

Table 3. Example of religious expressions present in all sides of the parallel corpus.

Expression	Meaning
علامات الساعة	Hereafter signs
سورة الكهف	Cave Surat
بسم الّله الرحمن الرحيم	In the name of Allah most gracious most merciful
الرسول عليه الصلاة و السلام	The prophet peace be upon him

In the same context, we noticed also that some conversations were very specific and concerned the area of studies. This is due to the fact that some records related to Annaba dialect were done by students who tended to discuss subjects related to courses and university in general. These discussions generated some expressions in the French language (which is used in university studies in Algeria for technical specialties).

These expressions have been translated to MSA and except for the two Algerian dialects texts, these expressions have been left in MSA since no equivalent dialectal words are available because of the poorness of these vernacular languages. We give in Table 4 some examples of these expressions.

5.3 Writing Rules

One of the most challenging issues related to Arabic dialects is their writing system. Since they were at a near time only spoken they do not have any orthography to respect. This issue is more complicated than we thought at the beginning of this work. In fact, for transcribing speech to text we decided to adopt a set of rules like writing words in Arabic if it is possible or writing them as they are uttered in the other case and transcribing non-Arabic phonemes like /*P*/ and /*G*/ as "پ" and "ڤ" respectively.

We noticed that these rules have not been fully observed. For instance, we found some dialectal expressions like قلتلك (I told you) or قَالتلو (she told him) which had to be written like in Arabic قلت لك and قَالت له in that order. Likewise, foreign letters rule has been applied only in ALG and ANB corpora, for other dialects these letters have not been used. An example of this is the word بورتَابل (mobile phone) from the Tunisian[2] corpus which is originally a french word, this word had to be written as it was uttered, that is پورتَابل. Another important point regards to writing dialect is the numerals. We did not impose any rules for them, they have been transcribed with letters. With a little hindsight, we see that it would have been better if we transcribed them with numerals instead of letters. This will have a positive impact on machine translation scores.

6 Conclusion

We presented our experiment in the area of building textual resources for Arabic dialects. We attempted to point out the main pitfalls to avoid when dealing with such a problem.

[2] Also widely used in Algeria and Morocco

Table 4. Some examples of french words existing in source dialect corpora and their translation in MSA and target dialect corpora.

Dialect/Language	Sentence	Meaning
MSA	الافراط المعرفي	Cognitive overload.
ALG	سورشارج كونيتيف	
ANB	سورشارج كونيتيف	
TUN	الافراط المعرفي	
MOR	الافراط المعرفي	
SYR	الافراط المعرفي	
PAL	الافراط المعرفي	
MSA	من المفروض جميع الَاختصَاصَات تدرس الخوَارزميَات	Algorithms should be taught in all specialties
ALG	نورمَالمون ليزوپسيون كَامل يقريوهم لَالقوريتميك	
ANB	نورمَالمون ليزوپسيون كل يقريوهم لَالقوريتميك	
TUN	من المفروض جميع الَاختصَاصَات تقرَا الحسَاب	
MOR	من المفروض كلشي الَاختصَاصَات تقري الخورزميَات	
SYR	لَازم كل الَاختصَاصَات تدرس الخوَارزميَات	
PAL	من المفروض جميع الَاختصَاصَات تدرس الخوَارزميَات	
MSA	المهم المسَابقة	The contest is the most important
ALG	الصح فَالكنكور	
ANB	الصح الكنكور	
TUN	المهم المسَابقة	
MOR	المهم المسَابقة	
SYR	المهم المسَابقة	
PAL	المهم المسَابقة	

First of all, Since MSA is the common point between dialects of Arab countries, we intuitively used it as a bridge between them (Arabic dialects). The influence of MSA on translators was apparent. The facility to keep certain words unchanged was a good option since we noticed that translating between so close languages was a very hard task according to all persons who participated in this operation.

Regards to topics, it is recommended to keep them far from specific domains like religion or scientific subjects. Indeed, Arab people when they address religious topics switch to MSA. This generates important MSA passages in dialectal corpora. In this respect and regards to the lexical poorness of Arabic dialects, domain-specific discussions where scientific and technical words are used have to be avoided.

In general, Arabic dialect vocabulary does not cover this area and people switch to standard languages like MSA, French and English. For writing Arabic dialects, we noticed that it was difficult to impose strict rules. People when writing dialect feel free to write how they want. In some way, this is natural. In dialects, there are no spelling mistakes since there are no rules.

References

1. Al-Sabbagh, R., Girju, R.: Mining the web for the induction of a dialectical arabic lexicon. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation, pp. 288–293 (2010)
2. Al-Sabbagh, R., Girju, R.: YADAC: Yet another dialectal arabic corpus. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation, pp. 2882–2889 (2012)
3. Almeman, K., Lee, M.: Automatic building of arabic multi dialect text corpora by bootstrapping dialect words. In: 1st International Conference on Communications, Signal Processing, and their Applications, pp. 1–6 (2013) doi: 10.1109/ICCSPA.2013.6487247
4. Bouamor, H., Habash, N., Oflazer, K.: A multidialectal parallel corpus of arabic. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation, pp. 1240–1245 (2014)
5. Boudlal, A., Lakhouaja, A., Mazroui, A., Meziane, A., Bebah, M., Shoul, M.: Alkhalil morpho sys1: A morphosyntactic analysis system for arabic texts. In: International Arab Conference on Information Technology (2010)
6. Boujelbane, R., BenAyed, S., Belguith, L. H.: Building bilingual lexicon to create dialect tunisian corpora and adapt language model. In: Proceedings of the Second Workshop on Hybrid Approaches to Translation, pp. 88–93 (2013)
7. Chen, S. F., Goodman, J.: An empirical study of smoothing techniques for language modeling. Computer Speech and Language, vol. 13, pp. 359–394 (1999) doi: 10.1006/csla.1999.0128
8. Cotterell, R., Callison-Burch, C.: A multi-dialect, multi-genre corpus of informal written arabic. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation, pp. 241–245 (2014)
9. Duh, K., Kirchhoff, K.: Lexicon acquisition for dialectal arabic using transductive learning. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, pp. 399–407 (2006)
10. Graff, D., Buckwalter, T., Jin, H., Maamouri, M.: Lexicon development for varieties of spoken colloquial arabic. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation, pp. 999–1004 (2006)
11. Hetzron, R.: The semitic languages (1997) https://books.google.dz/books?id=nbUOAAAAQAAJ
12. Maamouri, M., Bies, A.: Developing an arabic treebank: Methods, guidelines, procedures, and tools. In: Proceedings of the Workshop on Computational Approaches to Arabic Script-based languages, pp. 2–9 (2004)
13. Maamouri, M., Bies, A., Buckwalter, T., Diab, M., Habash, N., Rambow, O., Tabessi, D.: Developing and using a pilot dialectal arabic treebank. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation, LREC'06 (2006)
14. Maamouri, M., Bies, A., Kulick, S., Ciul, M., Habash, N., Eskander, R.: Developing an egyptian arabic treebank: Impact of dialectal morphology on annotation and tool development. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation, pp. 2348–2354 (2014)
15. Maamouri, M., Bies, A., Kulick, S., Zaghouani, W., Graff, D., Ciul, M.: From speech to trees: Applying treebank annotation to arabic broadcast news. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation, pp. 2117–2122 (2010)
16. Meftouh, K., Harrat, S., Jamoussi, S., Abbas, M., Smaili, K.: Machine translation experiments on PADIC: A parallel arabic dialect corpus. In: Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation, pp. 26–34 (2015)

17. Mubarak, H., Darwish, K.: Using twitter to collect a multi-dialectal corpus of arabic. In: Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing, pp. 1–7 (2014) doi: 10.3115/v1/W14-3601
18. Rytting, C. A., Zajic, D. M., Rodrigues, P., Wayland, S. C., Hettick, C., Buckwalter, T., Blake, C. C.: Spelling correction for dialectal arabic dictionary lookup. ACM Transactions on Asian Language Information Processing (TALIP), vol. 10, no. 3, pp. 1–15 (2011) doi: 10.1145/1929908.1929911
19. Zaghouani, W.: Critical survey of the freely available arabic corpora. In: Proceedings of the Workshop on Free/Open-Source Arabic Corpora and Corpora Processing Tools, LREC, pp. 1–8 (2014) doi: 10.13140/RG.2.1.1362.1284
20. Zbib, R., Malchiodi, E., Devlin, J., Stallard, D., Matsoukas, S., Schwartz, R., Makhoul, J., Zaidan, O. F., Callison-Burch, C.: Machine translation of arabic dialects. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 49–59 (2012)
21. Zribi, I., Khemakhem, M. E., Belguith, L. H.: Morphological analysis of tunisian dialect. In: International Joint Conference on Natural Language Processing, pp. 992—996 (2013)

ISSN 1870-4069

Combined Dictionary Approach to Opinion Analysis in Slovak

Martin Mikula[1], Xiaoying Gao[2],
Kristína Machová[1]

[1] Technical University of Košice,
Department of cybernetics and artificial intelligence,
Slovakia

[2] Victoria University of Wellington,
School of Mathematical and Computing Sciences,
New Zealand

{martin.mikula, kristina.machova}@tuke.sk
xiaoying.gao@ecs.vuw.ac.nz

Abstract. People produce more and more textual data every day. They speak with each other, write articles and comment on products and services. It is simple to analyze them manually, in case that we have a small amount of data. But when we have many data, it is very difficult to process them manually. We decided to use dictionary approach for the automatic analysis of comments in the Slovak language. The first algorithm achieved accuracy around 72%. One disadvantage was that the algorithm could not identify the polarity of all comments. More than 18% comments were not assigned polarity because they did not contain subjective words from the dictionary. The new approach combines the first dictionary approach with a probabilistic method which is used to create a new lexicon. The new dictionary was again used to analysis comments in the dataset. This new approach reduced the percentage of unidentified comments to 0.5%. The new approach outperformed the previous method and also achieved better results than Naïve Bayes classifier and Support Vector Machines (SVM) on the same dataset.

Keywords: Opinion analysis in Slovak, polarity of opinion, probability approach.

1 Introduction

People communicate through the Internet, talking about their feelings, comparing and rate the products and writing the blogs etc. All of these written texts are sources of very important information for people and companies who work with internet marketing and satisfaction surveys. If there are not a lot of data, it is easy to analyze them manually. When we have big volume of data, it is better to analyze them automatically. This is the main field of study of opinion classification.

Opinion classification is not a simple process. In this process, we try to identify the opinion of an author about a specific topic. An author is a person or a company, that has the specific opinion about the topic, who can also be called the opinion holder.

A topic means the object that the author talks about. It can be a product, a service or anything else. In some literature, opinion classification is called opinion mining. The opinion in opinion analysis consists of two parts. The first part is subjectivity/polarity. We know three basic types of subjectivity: positive, negative and neutral. The second part is the strength of this polarity. Some words can express polarity with higher strength than other, for example *the best* is more positive than *good*.

In this paper, we used the combination of two approaches to sentiment analysis. The first one is based on a dictionary. We created our own lexicon based on the English lexicon. This approach implements several functions (intensification, negation, stemming) to sentiment processing and classification. It achieved average results in the experiments. One of the main limitations was that the approach can not process around 18% of the comments. We decided to implement machine learning approach to classify these unclassified comments.

We chose multinomial Naïve Bayes classifier (NBC). It is a simple classifier based on probability theorem with good results for sentiment analysis. In the first phase, the combined approach used the lexicon approach to classify comments in the dataset. Then it split processed comments into two groups, classified and unclassified. The classified ones were used as the training dataset for Naïve Bayes classifier. In the last phase, the comments which were not classified by dictionary approach were classified by NBC. The final results of our combined approach outperformed other machine learning methods.

This paper is divided into five parts. The second part reviews the background and related work on sentiment analysis in English and Slovak language. In the third part, we described our combined which consists of the dictionary approach and the Naïve Bayes classifier. Experiments and results are in the fourth part. The last part is the conclusion and future work.

2 Background and Related Work

There are two basic types of methods for opinion analysis. The first type of methods are methods based on dictionaries. This method uses opinion words for text analysis. These words are stored in a sentiment lexicon. All words in the dictionary have been assigned polarity (positive or negative) and in some cases strength of polarity. In work by Hu and Liu [5], the authors used a dictionary created through a bootstrapping process using WordNet[1].

They counted the numbers of positive and negative words around the product feature. When there were more negative words than positive ones, then the final opinion was negative and otherwise positive. Benamara et al. [1] look for the best words, that can be used for sentiment analysis. The authors proposed three scoring methods based on adverb-adjective combinations (AACs).

The first scoring function was variable scoring (VS) which was used for adjectives. In the adjective priority scoring (APS) they selected a weight r which referred to the strength between adverb and adjective which it modified.

[1] http://wordnet.princeton.edu/

In the third scoring function, adverb first scoring (AFS) the authors used the same weight *r*, but it was applied to the adjective rather to the adverb. Their experiments show, that the APS achieved the best results which mean, that the adjectives are most useful ones for sentiment analysis. The influence of the other parts of speech was analyzed in work by Taboada et al. [14]. They created the dictionary manually. Their lexicon contained adjectives, adverbs, nouns and verbs. These types of words together achieved better results than individually.

The authors also studied the influence of different types of sentence processing. They analyzed the usage of two types of negation. The first was switch negation and the second was shift negation. The intensification was performed by using of percentage. The authors compared their lexicon with other lexicons such as General Inquirer, MPQA subjectivity lexicon, SentiWordNet etc. These dictionaries were compared on the dataset obtained from Epinion.com. The approach presented in this work which included all types of words, shift negation and intensification with other features achieved the best results.

An application for opinion classification in the Slovak language based on dictionaries is described in Machová and Krajč [6]. This application used dictionaries of positive and negative words. It was able to process intensification and negation. The range of influence of intensification and negation was determined by a dynamic coefficient. This coefficient was set as static at the beginning, but it was calculated base on sentence length later. This application achieved 86.2% precision for positive and 69.2% for negative comments.

The second type of methods are approaches based on machine learning. These approaches use machine learning methods for opinion classification. The most common methods are Naïve Bayes classifier, Support Vector Machines (SVM) and Maximum Entropy. An SVM as machine learning method was used in work [9]. They trained the SVM with linear kernel on various features such as N-grams, number of all-capitalized words, POS tags, polarity dictionaries, punctuation marks, emoticons, lengthened words, clustering and negation. From N-grams, from 1- to 4-grams were considered. The authors used five different polarity dictionaries, three were manually created and two created automatically.

They also used non-single punctuation marks (e.g. !! or !?) and words with a letter repeating more than twice. Negation was applied from the begging of a negation to the first punctuation. Their approach obtained 69.02% macro F-measure. A maximum entropy-based classifier was used in the work Proisl et al. [12]. They used N-grams, the length of tweet, polarity dictionary, emoticons and abbreviations and negation as a feature set. The authors applied only unigrams and bigrams. In contrast to previous approach, the frequency-weighted N-grams were used. The improved AFINN-111 lexicon [10] was implemented to this approach. The authors added 343 new words to this lexicon.

Also a list of emoticons and abbreviations was used. Negation was applied only on the next three words. This approach achieved 63.06% macro F-measure. The work by Habernal et al. [3] is dedicated to the sentiment analysis in Czech language. The authors used several pre-processing methods such as stemming, POS tagging, misspellings and grammatical corrections.

Table 1. Examples of words stored in the lexicon.

Word	Strength of polarity	Polarity
zlý (bad)	-1	n
dobrý (good)	1	p
najlepší (best)	3	p
veľmi (very)	2.0	i
dosť (enough)	1.5	i
nebol (wasn't)	-1	o
trochu (litte)	1.25	i

They experimented with five feature selection methods (N-grams, character N-gram features, POS-related features, emoticons and TFIDF variants). Their methods were tested with two classifiers: SVM and Maximum entropy. The best results were achieved by the maximum entropy using all types of features. This method was the best on all tested datasets.

In some approaches, the researchers use the two or more types of methods together and combine them. Sindhwani and Melville [13] combined the unsupervised and the semi-supervised learning method. The unsupervised method was based on the dictionary which contained 2986 human labeled words. This method was utilized for domain adaptation. After this step, the semi-supervised algorithm was used for lexical classification.

The bipartite representation of data (document-word bipartite) was used as the semi-supervised method. They used Regularized Least Squares (RLS) as the classification algorithm because they had sparse data. The prior knowledge of sentiment words was incorporated into the model by lexical RLS. This approach was tested on three different domains. The lexical RLS was compared with semi-supervised lexical RLS and achieved better results. The combination of four different approaches was used in the work Hagen et al. [4].

The authors reimplemented following methods: 1. an SVM classifier from work Mohammad et al. [9], 2. a stochastic gradient descent classifier [2], which was trained on unigrams, stems, clustering polarity dictionary and negation, 3. a maximum entropy-based classifier from work [12] and 4. a logistic regresion [8] as a supervised machine learning was used. It was trained on POS tags, N-grams, and polarity dictionaries. The final decision was made on the average probability (the confidence score) of each classifier for each of the three classes. This ensemble achieved F1-score 64.84%.

3 Our Combined Approach

In our approach, we first used the dictionary approach for sentiment analysis in Slovak. We created the lexicon which contained 1430 words. It achieved accuracy around 72%. But more than 18% comments were not classified because they did not contain words from the dictionary. This was the reason that we tried to implement a probability approach to generating a new lexicon. This lexicon was generated from comments, that were labeled by the previous dictionary based approach. Then we used this new lexicon for labeling the corpus again.

3.1 Dictionary Approach

Our lexicon was created by translating from an English lexicon used in work Hu and Liu [5]. The original lexicon contained 6789 words, 4783 negative and 2006 positive words. We used Slovak thesaurus for searching synonyms to every word which we translated. The Slovak lexicon contained 598 positive words, 772 negative words, 41 intensifiers and 19 negations. Every word in the dictionary had been assigned *polarity*, and *strength of polarity*. We used 4 types of polarity:

- p - positive word,
- n - negative word,
- i - intensifier,
- o - opposite.

For polarity strength, we decided to use a range from -3 (the most negative) to +3 (the most positive). The words were stored in two forms in the lexicon. If it was possible, the word was saved in the edited form. We could edit words which finish with vowels: ”a”, ”e”, ”i”, ”u”, ”y”, in their basic form. Other words were saved in a form without editing. Examples of words in the lexicon are in Table 2.

Our algorithm for sentiment analysis worked in several phases. In the pre-processing phase, it edited the incoming text. It removed diacritics, changed all letters to lowercase and edited words with the Lancaster stemming algorithm[2]. The algorithm split the text into sentences and words. Every word from comment was compared with words in the lexicon.

If the algorithm found the word in the dictionary, it updated the polarity of the sentence by adding the strength of polarity of the word. We implemented two functions (intensification and negation) to text processing into this algorithm. The first function was intensification. Our intensifier had an assigned strength of polarity from 1.0 to 2.0. This type of intensification allowed us to assign higher intensity for words with high strength of polarity and lower intensity for words with the small strength of polarity. For example, if we had a combination:

- Veľmy dobrý (very good) where *very* is intensifier with value 2.0 and *good* has value +1 $\rightarrow$ the phrase had value of polarity $1\text{x}2.00 = 2$,
- Dosť dobrý (enough good) where *enough* is intensifier with value 1.5 and *good* has value +1 $\rightarrow$ the phrase had value of polarity $1\text{x}1.5 = 1.5$.

The second function was a negation. The algorithm was able to process two types of negation by using combined negation processing [7]. The negation is processed by the following rules:

if word is opposite **then**
    search the first word with polarity
    if strength of polarity is +/-1 or +/-2 **then**
        use switch negation

[2] http://web.archive.org/web/20140725115219/http://www.comp.lancs.ac.uk/computing/research/stemming/index.htm

```
        else
            if strength of polarity is +/-3 or intensifier then
                use shift negation
            end if
        end if
    end if
```

The switch negation changes the value of the word polarity to the opposite with the same strength. The shift negation shifts the strength of polarity to the opposite way by an exact value. We used two/negative two as value for shift negation in our algorithm. Here are a few examples of our implementation of negation.

1. Switch negation: Film nebol dobrý. (The movie wasn't good.) $\rightarrow$ *wasn't* is opposite and *good* has value +1 $\rightarrow -1\text{x}1 = -1$.
2. Shift negation: Film nebol najlepší. (The movie wasn't the best.) $\rightarrow$ *wasn't* is opposite and *the best* has value +3 $\rightarrow -2 + 3 = -1$.
3. Shift negation: Film nebol veľmi dobrý. (The movie wasn't very good.) $\rightarrow$ *wasn't* is opposite, *very* is intensifier with value 2.0 and *good* has value +1 $\rightarrow -2+2.0\text{x}1 = 0$.

The described approach achieved accuracy around 72%. The big problem was that this version of the algorithm and the actual dictionary were not able to classify more than 18% comments in the dataset.

3.2 Probability Method

In order to classify comments which did not contain words from the dictionary, we decided to use a method which created a new dictionary from comments that had been classified by the dictionary approach. This method divided classified comments into the positive and negative group. These groups were sorted from the most positive/negative to the less positive/negative.

Then the two groups were compared to discover which one contained fewer comments. We join the two datasets together and form a new dataset. A new joined dataset was created using all comments in the smaller group and the same number of samples were chosen from the bigger group, so the new dataset is balanced. This method created a training dataset which contained 50% of positive and 50% of negative comment. This distribution was very important because, if we had more comments with one polarity, it could influence the results.

The training dataset was used to generate a new dictionary. The algorithm splits all comments in training dataset to sentences and words. Then it counted term frequency depending on the presence of each word in positive and negative comments. The new lexicon was based on term frequency of each word connected with each class (positive or negative).

Then if the word from dictionary was found in a new comment, the probability *P*, that this word *w* is from class *c* was computed by the simple probability method described in formula 1:

$$P(w_c) = \frac{\sum w_c}{\sum w}, \tag{1}$$

where:
$P(w_c)$ - the probability that the word is from class *c*,
$\sum w_c$ - the number of occurrences of word *w* in class *c*,
$\sum w$ - the number of occurrences of word *w* in the whole dataset.

In case that the word was not connected with a specific class and the probability would be zero, we implemented a method which returned a very low number instead of zero.

The value of polarity of the new comment consists of probabilities of each word connected to the positive and negative class. We computed the probability by formula 2:

$$P(sentence_c) = \frac{\sum P(w_c)}{\sum w}, \tag{2}$$

where:
$P(sentence_c)$ - the probability that sentence is from class *c*,
$\sum P(w_c)$ - the summed probabilities of each word from the sentence which is from class *c*,
$\sum w$ - the number of words in sentence.

The new comment was added to the class which had bigger final probability. The comment was positive if the positive probability was bigger than negative probability and vice-versa.

Whole process can be described by following:

classify the dataset by the dictionary approach
for all all comments in the dataset **do**
    if comment has label **then**
        add comment to the positive or the negative subset
    end if
end for
if size of the positive subset $>$ size of the negative subset **then**
    add to the training set whole negative subset and same number of comments from the positive subset
else
    add to the training set whole positive subset and same number of comments from the negative subset
end if
train classifier
for all all comments in the dataset **do**
    classify comment
end for

This new approach reduced the number of unclassified comment to 0.5%.

Table 2. Opinion classification using dictionary approach (DA).

Approach	F1(+)	F1(-)	Macro F1
DA with unclassified comments	0.742	0.645	0.694
DA without unclassified comments	0.89	0.824	0.857

Table 3. The comparison of different approaches for opinion analysis.

Approach	F1(+)	F1(-)	Macro F1
Our dictionary approach	0.742	0.645	0.694
Our combined approach	**0.872**	**0.863**	**0.868**
Naïve Bayes	0.811	0.812	0.812
SVM	0.845	**0.863**	0.854

4 Experiments and Results

The proposed approach was tested on a dataset with 5242 comments. The dataset contains 2573 positive and 2669 negative comments from different areas (politics decisions, electronics and books reviews, movie reviews, etc.). The comments contain 182 645 words. The neutral comments were removed.

The metrics which we used to evaluate our method were based on *precision* and *recall* to obtain *F1* measure per class (positive and negative). *F1* is the harmonic mean between precision and recall. Some datasets are unbalanced, so we decided to use *Macro-F1* that is computed by calculating *F1* values for each class, and then averaging over all classes. Thus, *Macro-F1* shows the effectiveness in each class, independently of the size of the class.

In our first experiment, we tested the dictionary approach. We present two set of results in Table 4. The first row shows the results with unclassified comments where the unclassified ones were treated as wrong results. The second row is the result with the only classified ones.

These results show that the unclassified comments degrade the performance. The accuracy of results with unclassified comments was around 72%, and without unclassified comments around 86%. This version of the algorithm was not able to classify 976 comments which are around 18%. The unclassified comments constituted 65% of the incorrectly categorized positive comments and 60% of the incorrectly categorized negative comments. This was the reason why we decided to create this combined approach.

In our next experiment, we compared our combined approach with other approaches. It was compared with previous dictionary approach and with two machine learning methods: Naïve Bayes classifier and SVM. The two machine learning methods (NB and SVM) are implemented in data mining tool RapidMiner[3] and used 10 fold cross-validation to got the results.

The results show that our combined approach achieved better results than the original dictionary approach and the Naïve Bayes classifier. The proposed combined approach highly outperformed the original dictionary based approach. The combined approach can classify 5226 comments from the dataset which represents 99.7%.

[3] https://rapidminer.com/

Table 4. The comparison of the dictionary approach (DA) and the combined approach (CA) on the different datasets.

Approach	F1(+)	F1(-)	Macro F1
DA on our dataset	0.742	0.645	0.694
CA on our dataset	0.872	0.863	0.868
DA on movie review	0.727	0.611	0.669
CA on movie review	0.741	0.599	0.670

The combined approach also achieved significantly better results than the Naïve Bayes classifier in all measures. This approach obtained slightly better results than SVM. In our next experiment, we compared our approach with a standard movie review dataset used in work Pang and Lee [11]. This dataset includes 1,000 positive and 1,000 negative movie reviews annotated by the human annotator. They were collected from rottentomatos.com. All text was translated to the Slovak language and converted to lowercase.

From the results in table 4 can be seen, that combined approach was better on our dataset. It was also slightly better on the standard dataset. The analysis on movie dataset was also compared with the results from the work by Taboada[14].

They achieved performance between 68.05% (a simple sentiment analysis using just words from the dictionary) and 76.37% (a sentiment analysis based on additional features). Our combined dictionary achieved slightly worse results which can be caused by misunderstanding during automatic translation process.

5 Conclusion

In this paper, we proposed a new combined approach to opinion analysis. It was created by the combination of a dictionary approach and a probability method. The dictionary approach had analyzed unlabeled data and classified comments into the positive and the negative group. But it can not analyze around 18% comments.

They were not labeled because these comments did not contain words from the dictionary. For this reason, we implemented a probabilistic approach which created new dictionary based on comments analyzed by the dictionary approach. The algorithm found new polarity words, typical for this dataset and added them to a new dictionary.

The dataset was again analyzed with this new dictionary and more than 99.6% comments were classified. This combined approach achieved better results than the previous dictionary approach and the Naïve Bayes classifier. It was compared also with SVM with which achieved comparable results. In the future research, we want to use this combined approach for domain adaptation and active learning in the data stream.

If this approach will be used in active learning, the dictionary approach can create training dataset. This dataset will be used for training classifier and the trained classifier will be then used for labeling other data from the stream. In case that the domain will be changed, the algorithm will use this combined approach for creating new training dataset, especially for this new domain. The machine learning algorithm can be retrained on this new training dataset.

Acknowledgments. The work presented in this paper was supported by the Slovak Grant Agency of the Ministry of Education and Academy of Science of the Slovak Republic under VEGA grant No. 1/0493/16 (70%) and by the European Commission through the Thelxinoe project (30%). This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

References

1. Benamara, F., Cesarano, C., Picariello, A., Reforgiato, D., Subrahmanian, V. S.: Sentiment analysis: Adjectives and adverbs are better than adjectives alone. In: Proceedings of the International Conference on Weblogs and Social Media (2007)
2. Günther, T., Furrer, L.: GU-MLT-LT: Sentiment analysis of short messages using linguistic features and stochastic gradient descent. In: Proceedings of the Seventh International Workshop on Semantic Evaluation, vol. 2, pp. 328–332 (2013)
3. Habernal, I., Ptáček, T., Steinberger, J.: Supervised sentiment analysis in Czech social media. Information Processing and Management, vol. 50, no. 5, pp. 693–707 (2014) doi: 10.1016/j.ipm.2014.05.001
4. Hagen, M., Potthast, M., Büchner, M., Stein, B.: Webis: An ensemble for twitter sentiment detection. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval '15), pp. 582–589 (2015) doi: 10.18653/v1/S15-2097
5. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 168–177 (2004) doi: 10.1145/1014052.1014073
6. Machová, K., Krajč, M.: Opinion classification in threaded discussions on the web. In: 10th Annual International Conference Znalosti, pp. 136–147 (2011)
7. Mikula, M., Machová, K.: Spracovanie negácie pre klasifikáciu názorov v slovenskom jazyku. In: Data a znalosti, pp. 41–45 (2015)
8. Miura, Y., Sakaki, S., Hattori, K., Ohkuma, T.: TeamX: A sentiment analyzer with enhanced lexicon mapping and weighting scheme for unbalanced data. In: Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval '14), pp. 628–632 (2014) doi: 10.3115/v1/S14-2111
9. Mohammad, S. M., Kiritchenko, S., Zhu, X.: NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In: Proceedings of the Seventh International Workshop on Semantic Evaluation, pp. 321–327 (2013) doi: 10.48550/arXiv.1308.6242
10. Nielsen, F. Å.: A new ANEW: evaluation of a word list for sentiment analysis in microblogs. pp. 93–98 (2011) doi: 10.48550/arXiv.cs/0409058
11. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (2004) doi: 10.3115/1218955.1218990
12. Proisl, T., Greiner, P., Evert, S., Kabashi, B.: KLUE: Simple and robust methods for polarity classification. In: Proceedings of the Seventh International Workshop on Semantic Evaluation, pp. 395–401 (2013)
13. Sindhwani, V., Melville, P.: Document-word co-regularization for semi-supervised sentiment analysis. In: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, pp. 1025–1030 (2008) doi: 10.1109/ICDM.2008.113
14. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. Computational linguistics, vol. 37, no. 2, pp. 267–307 (2011) doi: 10.1162/COLI_a_00049

ISSN 1870-4069

Citation-based Prediction of Birth and Death Years of Authors

Dror Mughaz[1,2], Yaakov HaCohen-Kerner[2], Dov Gabbay[1,3]

[1] Bar-Ilan University,
Dept. of Computer Science,
Israel

[2] Lev Academic Center,
Dept. of Computer Science,
Israel

[3] Kings College London,
Dep. of Informatics,
United Kingdom

myghaz@gmail.com, kerner@jct.ac.il,
dov.gabbay@kcl.ac.uk

Abstract. This paper presents an unusual approach in text mining and feature extraction for identifying the era of anonymous texts that can help in the examination of forged documents or extracting the time-frame of which an author lived. The work and the experiments concern rabbinic documents written in Hebrew, Aramaic and Yiddish texts. The documents are undated and do not contain any bibliographic sections, which leaves us with an interesting challenge. This study proposes a few algorithms based on keyphrases that enable prediction of the time-frame of which the authors lived based on the temporal organization of references using linguistic patterns. Based on the keyphrases and the citations we formulated various types of "Iron-clad", Heuristic and Greedy constraints defining the birth and death years of an author that lead to an interesting classification task. The experiments that were applied on corpora containing texts authored by 12, 24 and 36 rabbinic authors show promising results.

Keywords: Citation analysis, text mining, Hebrew-Aramaic documents, knowledge discovery, time analysis, undated citations, undated documents.

1 Introduction

Extracting the era of a book/manuscript and determining who the author who wrote it, is very challenging and essential problems, the use of citations can help us to do that. Citations provide a lot of important information to studies in different areas such as academic, legal and religious. Therefore, extracting and analyzing citations

automatically is growing and gaining momentum. Search engines and digitized corpora enable identification and extraction of citations. Hence, citation analysis has an increased importance.

The use of citations is not limited to academic papers it is used also in rabbinic responsa (questions and detailed rabbinic answers). The citations that includes in rabbinic texts are more hard to define and to pull out than citations in academic papers because: (1) There is no bibliography at the end of a responsa; (2) The complex morphology of Hebrew, Aramaic and Yiddish; (3) NLP in Hebrew Aramaic and Yiddish has been little studied; (4) Many citations in Hebrew-Aramaic-Yiddish documents are ambiguous; (5) these citations are undated and (6) Plenty of different styles and syntactic used to display citations [1].

The Semitic languages are substantially different from the Latin languages. Due to that, Hebrew, which is Semitic language, will have very different processing from English in a variety of linguistic aspects. In Hebrew the writing direction is from right-to-left which is different from the Latin languages. Also, in Hebrew there are no vowels thus there are lots of ambiguous words. For example, the word דלק can be read as Delek, which means a fuel or Dalak, which means burned and as well to-chase; another example is the word ספר, which can be can be read as Safar, which means counted, or Sapar, which means barber, or Sefer which means a book [2].

The morphology of Hebrew, compared to the morphology of English, is plentiful. Most the function words in English are affixes in Hebrew. The Hebrew has a normal form, which called a "root" which has many different forms that sometimes not just add characters to it but also change the structure of the root pattern and the meaning of the word. (For example, לשמש = ל+שמש (La-Shemesh, which means to the sun (NP)) or לשמש (Leshamesh, which means to serve (adv))) [2].

There is relatively a high rate of acronyms and abbreviations in Hebrew texts. The number of abbreviations is about 17,000, compared to 40,000 lexical entries in Hebrew [3] it is relatively high, hence, another type of ambiguity. In Hebrew Rabbinic texts the use of acronyms and abbreviations are by far more plentiful than regular Hebrew texts.

HaCohen-Kerner et al. [3, 4, 5, 6] construct a system to disambiguate Hebrew and Aramaic acronyms for classical Jewish texts, mostly in pre-Modern Hebrew. They used an existing dictionary of acronym expansions and with the use of machine learning techniques they achieved high accuracy of automatic acronym expansion.

HaCohen-Kerner et al. [7] showed that manual disambiguation of an acronym is greatly time-consuming and even for highly trained human experts it's a challenging task.

Liebeskind et al., [8; 9;10] addressed the task of thesaurus construction, aiming to enable potential users of the Responsa Project to search for modern terms and obtain semantically related terms from earlier periods in history.

They proposed an algorithmic scheme for generating a co-occurrence based thesaurus in Morphologically Rich Languages (MRL) and demonstrate its empirical benefit for the Hebrew diachronic thesaurus [10]. Then, they introduced a semi-automatic iterative Query Expansion (QE) scheme that increases recall and the effectiveness of lexicographer manual effort [8]. Later, their scheme was extended to deal with Multi-Word Expressions [9].

This research uses undated citations of other dated authors in order to assess the date of undated documents. The assessment based on a several rules of different level of

certainty: "iron-clad", heuristic and greedy. The rules are based on citations: (I) typical citations with no cue words and (II) citations with cue words such as: "late" ("of blessed memory"), "friend" and "rabbi".

Previous studies in this area [11, 12, 13] were made on: 12 or 24 authors, on two different years' intervals and without normalizing the difference between those intervals. We expend these studies from several aspects: using a much larger corpus, increasing the number of authors, normalizing the amplitudes years (to have a comparison) and examine the constants that are part of the heuristic rules.

This paper is organized as follows: Section 3 presents various constraints of different degree of certainty: "iron-clad", heuristic and greedy constraints that are used to estimate the birth and death years of authors. Section 4 describes the model. Section 5 introduces the tested dataset, the results of the experiments and their analysis. Section 6 summarizes, concludes and proposes future directions.

2 Related Works

The first to propose extraction of citation automatically for indexing and analysis from academic corpora papers was Garfield [14]. Berkowitz and Elkhadiri [15] pull out titles and author names from articles. Giuffrida et al. [16] extract author names from metadata of computer science articles by knowledge-based framework. Seymore et al. [17] extract author name by hidden Markov models from a small corpus of computer science articles.

Teufel et al. [18] classify citations to their function by extraction automatically the citations and their context (the reason for citing the paper). Tan et al. [19] disambiguate author of the results of automatically-crafted web searches. Improvement of extracting terms using citations has been done. Bradshaw [20] uses a fixed window round citations to extract terms. Ritchie et al. [21] improve retrieval effectiveness by selecting text from around citations in order to extract good index terms.

Temporal citation-related problems were done on the traditional Western scientific literature. Popescul et al. [22] present an approach for identify and cluster temporal deal with hyper-linked scientific texts databases. Kolomiyets et al. [23] and Bethard et al. [24] dealt with the issue of creating a timeline, the timeline is for each text on its own and relating to that text they create chronological order. Schwartz et al. [25] and Wen et al. [26] studied texts on social networks addressing a story/documentation of the specific process (medical, etc.) that occurs to individual.

Many studies in information retrieval have been done on citations (e.g. IR [27; 28; 21; 29; 30; 15; 2]). However, our study we are addressing much harder issue of citations that included in rabbinic texts. Mughaz et al. [13] present's approach of cross generation and citation-based method to date undated authors. Their experiment was based on a small corpus of only 12 authors and 24 authors.

3 Citation-Based Constraints

In this part we show the citation-based formulated rules in order to estimate the birth and death years of a writer X (the outcomes point to particular years) based on his texts

and on different writers' (Yi) texts who mention X or one of his books. We are assuming that the death years and birth years of all writers (Yi) are known, but those of the examined writer (X). Now we are giving constants and notions: X – The author under examination, Yi – Other writers, B – Birth year, D – Death year, MIN – Minimal age (currently 30) of a rabbinic writer when he starts to write his response, MAX – Maximal life period (currently 100) of a rabbinic author and RABBI_DIS – The age gap between rabbi and his scholarly student (currently 20).

The assessment of MIN, MAX, RABBI_DIS constants are heuristic but they are reasonable to a typical responsa authors' way of living. There are several types of citations: general citations without any cue word and citations with cue words, as: "Rabbi", "Friend", and "Late" ("of blessed memory"). The citations are divided to two kinds: those who citing living authors and those who citing dead authors. Unlike academic papers, the responsa contain much more citations to dead authors than to living authors.

We will introduce citation-based constraints of different degrees of certainty: "iron-clad" (I), heuristic (H) and greedy (G). "Iron-clad" constraints are absolutely true, without any exception. Heuristic constraints are almost always true. Exceptions can occur when the heuristic estimates for MIN, MAX and RABBI_DIS are incorrect. Greedy constraints are rather reasonable constraints for responsa authors. However, sometimes wrong estimates can be drawn while using these constraints. Each constraint will be numbered and its degree of certainty will be presented in brackets.

3.1 "Iron-Clad" and Heuristic Constraints

First of all, we present two general heuristic constraints based on authors that cite X, which are based on regular citations (i.e., without mentioning special cue words, e.g., "friend" and "rabbi").

General constraint based on authors that were cited by X:

$$D(X) >= MAX(B(Yi)) + MIN \quad (1\ (H))$$

X must be alive when he cited Yi, so we can use the earliest possible age of publishing of the latest born author Y as a lower estimate for X's death year.

General constraint based on authors that cite X:

$$B(X) <= MIN(D(Yi)) - MIN \quad (2\ (H))$$

All Yi must have been alive when they cited X, and X must have been old enough to publish. Therefore, we can use the earliest death year amongst such authors Yi as an upper estimate of X's earliest possible publication age (and thus his birth year).

General constraints based on references to year Y that were cited by X, later we will address it as a "years-feature":

$$D(X) >= MAX(Y) + MIN\ (3\ (H))$$

X must be alive when he mentioned the year Y, we can use the most recent year referred by X to evaluate the death year of X as estimation for X's death year.

Because writer, in a lot of cases dos not write until his “last day” we add heuristic constant “MIN”.

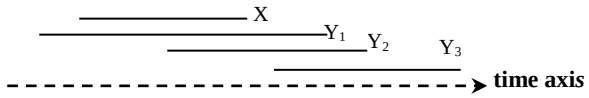


Fig. 1. Citations mentioning X as "late".

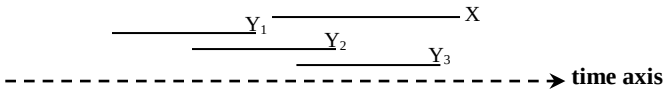


Fig. 2. Citations by X who mentions others as "late".

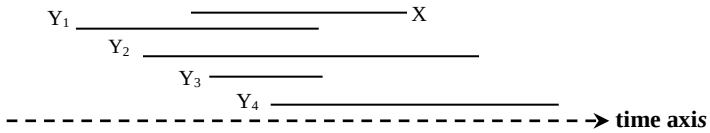


Fig. 3. Citations by authors who refer to X as their Friend/Rabbi.

Posthumous citation constraints

Posthumous constraints estimate the birth and death years of an author X based on citations of authors who refer to X as "late" ("of blessed memory") or on citations of X who mentions other authors as "late". Fig. 1 describes possible situations where various kinds of authors Yi (i=1, 2, 3) refer to X as "late". The lines depict authors' life spans where the left edges represent the birth years and the right edges represent death years. In this case (as all Yi refer to X as "late"), we know that all Yi died after X, but we do not know when they were born in relation to X's birth. Y1 was born before X's birth; Y2 was born after X's birth but before X's death; and Y3 was born after X's death:

$$D(X) <= MIN(D(Yi)) \quad (4\ (I))$$

However, we know that X must have been dead when Yi cited him as "late", so we can use the earliest born such Y's death year as an upper estimate for X's death year. Like all authors, dead authors of course have to comply to constraint (2) as well.

Now look at the cases where the author X, we are studying refers to other authors Yi as "late". Fig 2 describes possible situations where X refers to various kinds of authors Yi (i = 1, 2, 3) as "late". All Yi died before X's death (or maybe X is still alive). Y1 died before X's birth; Y2 was born before X's birth and died when X was still alive; and Y3 was born after X's birth and died when X was still alive:

$$D(X) >= MAX(D(Yi)) \quad (5\ (I))$$

X must be alive after the death of all Yi who were cited as "late" by him. Therefore, we can use the death year of the latest-born such Y as a lower estimate for X's death year.

$$B(X) >= MAX(D(Yi)) - MAX \quad (6\ (H))$$

X may be born after the death year of the latest-dying person, who X wrote about. Thus, we use the death year of the latest-born such Y minus his max life-period as a lower estimate for X's born year.

Contemporary citation constraints

Contemporary citation constraints calculate the upper and lower bounds of the birth year of an author X based only on citations of known authors who refer to X as their friend/rabbi. This means there must have been at least some period in time when both were alive. Fig 3 describes possible situations where various kinds of authors Yi refer to X as their friend/rabbi. Y1 was born before X's birth and died before X's death; Y2 was born before X's birth and died after X's death; Y3 was born after X's birth and died before X's death; and Y4 was born after X's birth and died after X's death. Like all authors, contemporary authors of course have to comply to constraints 1 and 2 as well:

$$B(X) >= MIN(B(Yi)) - (MAX - MIN) \quad (7\ (H))$$

All Yi must have been alive when X was alive, and all of them must have been old enough to publish. Therefore, X could not be born MAX-MIN years before the earliest birth year amongst all authors Yi:

$$D(X) <= MAX(D(Yi)) + (MAX - MIN) \quad (8\ (H))$$

Again, all Yi must have been alive when X was alive, and all of them must have been old enough to publish. Thus, X could not be alive MAX-MIN years after the latest death year amongst all authors Yi.

3.2 Greedy Constraints

Greedy constraints bounds are sensible in many cases, but can sometimes lead to wrong estimates.

Greedy constraint based on authors who are mentioned by X:

$$B(X) >= MAX(B(Yi)) - MIN \quad (9\ (G))$$

Many of the citations in our research domain relate to dead authors. Thus, most of the citations mentioned by X relate to dead authors. That is, many of Yi were born before X's birth and died before X's death. Therefore, a greedy assumption will be that X was born no earlier than the birth of latest author mentioned by X; but because that may be at least one case where Y was born after that X was born so we subtract MIN.

Greedy constraint based on references to year Y that were cited by X, later we will address it as a "years-feature":

$$B(X) >= MAX(Y) - MINy \quad (10\ (G))$$

X reminds years he usually writes the current year in which he wrote the document or several years before. Most of the time the maximum year, Y, reduces MIN is larger than X's born year because of that the “MIN” in (10 (G)) cannot be as the “MIN” in the other constraint so is “MINy” (currently 60).

Greedy constraint based on authors who refer to X:

$$D(X) <= MIN(D(Yi)) - MIN \quad (11\ (G))$$

As mentioned above, most of the citations mentioned by Yi relate to X as dead. Therefore, most of Yi die after X's death. Therefore, a greedy assumption will be that X died no later than the death of the earliest author who refers to X minus MIN.

Constraints refinements 9-11 are presented by constraints 12-17. Constraints 12-14 are due to X citing Yi and Constraints 15-17 are due to Yi citing X.

Greedy constraint for defining the birth year based only on authors who were cited by X as "late":

$$B(X) >= MAX(D(Yi)) - MIN \quad (12\ (G))$$

When taking into account only citations that are cited by X, most of the citations, relate to dead authors. That is, most of Yi died before X's birth; in addition, an author doesn't write from his birth but usually until near his death. Therefore, a greedy assumption will be that X was born no earlier than the death of the latest author mentioned by X minus MIN.

Greedy constraint for defining the birth year based only on authors who are mentioned by X as a "friend":

$$B(X) <= MIN(B(Yi)) + RABBI\_DIS \quad (13\ (G))$$

When taking into account only citations that are mentioned by X, which relate to contemporary authors, a greedy constraint can be that X was born no later than the birth of the earliest author mentioned by X as a friend. Because many times older author is mentioning young author as a friend we need to add RABBI_DIS.

Greedy constraint for defining the birth year based only on authors who are mentioned by X as a "rabbi":

$$B(X) <= MIN(B(Yi)) + RABBI\_DIS \quad (14\ (G))$$

When taking into account only citations that are mentioned by X, which relate to contemporary authors, a greedy constraint can be that X was born not later than the birth of the earliest author mentioned by X as a RABBI. Because of the age difference between the rabbi and his student is about 20 years we need to add RABBI_DIS.

Greedy constraint for defining the death year of X based only on authors who cited X as "late":

$$D(X) <= MIN(B(Yi)) + MIN \quad (15\ (G))$$

When taking into account only citations that are mentioned by Yi who relate to X as "late", a greedy assumption can be that X died no later than the birth of the earliest author who cited X as "late" and because author doesn't writes from his birth we need to add MIN.

Greedy constraint for defining the death year of X based only on authors who cited X as a "friend":

$$D(X) >= MAX(D(Yi)) - RABBI\_DIS \quad (16\ (G))$$

When taking into account only citations that are mentioned by Yi who cited X as a friend, all Yi must have been alive when X was alive, and all of them must have been old enough to publish and many times older author is mentioning young author as a friend but the opposite never happen. Therefore, a greedy assumption will be that X

died no earlier than the death of the latest author who cited X as a friend minus RABBI_DIS.

Greedy constraint for defining the death year of X based only on authors who cited X as a "rabbi":

$$D(X) >= MAX(D(Yi)) - RABBI\_DIS \quad (17\ (G))$$

It is the same principle as the constraint for defining the born year but because hear the student mention the rabbi we need to reduce RABBI_DIS.

3.3 Birth and Death Year Tuning

Application of the Heuristic and Greedy constraints can lead to anomalies, such as an author's decease age being unreasonably old or young. Another possible anomaly is that the algorithm may yield a death year greater than the current year (i.e. 2014). Therefore, we added some tuning rules: D – death year, B – born year, age = D-B.

Current Year: if (D > 2016) {D = 2014}, i.e., if the current year is 2014 the algorithm must not give a death year greater of 2014.

Age: if (age > 100) {z = age – 100; D = D – z/2; B = B + z/2} if (age < 30) {z = 30 – age; D = D + z/2; B = B - z/2}. Our assumption is that an author lived at least 30 years and no more than 100 years. Thus, if the age according to the algorithm is greater than 100, we take the difference between that age and 100, then we divide that difference by 2 and normalize D and B to result with an age of 100.

4 The Model

The main steps of the model are presented below. Most of these steps were processed automatically, except for steps 2 and 3 that were processed semi-automatically.

1 **Cleaning the texts**. Since the responsa may have undergone some editing, we must make sure to ignore possible effects of differences in the texts resulting from variant editing practices. Therefore, we eliminate all orthographic variations.

2 **Normalizing the citations in the texts**. For each author, we normalize all kinds of citations that refer to him (e.g., various variants and spellings of his name, books, documents and their nicknames and abbreviations). For each author, we collect all citation syntactic styles referred to him and then replace them to a unique string.

3 **Building indexes**, e.g., authors, citations to "late"/friend/rabbis and calculating the frequencies of each item.

4 **Citation identification** into various categories of citations, including self- citations.

5 **Performing various combinations of "iron-clad" and heuristic constraints** on the one hand, **and greedy constraints** on the other hand, **to estimate** the birth and death years for each tested author.

6 **Calculating averages** for the best "iron-clad" and heuristic version and the best greedy version**.**

Table 1. Birth average distance. without years-feature.

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	Age	const	Age	Age
		46.56	60.97	37.5	**26.32**
	24	Age	const	Age	Age
		34.93	33.65	28.58	**26.13**
	36	Age	const	Age	Age
		31.47	27.63	23.39	**19.6**
Greedy	12	Age	Age	const	Age
		30.22	39.85	**25.28**	28.37
	24	Age	Age	Age	Age
		15.28	19.98	17.46	15.71
	36	Age	Age	Age	Age
		12.46	20.51	14.24	14.4

Table 2. Death average distance. without years-feature.

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	Age	no tuning	const	Age
		43.1	41.5	45.1	**28.3**
	24	Age	no tuning	const	Age
		23.98	**19.9**	32.7	24
	36	Age	no tuning	Age	Age
		17.72	18.2	20	17
Greedy	12	Age	Age	Age	Age
		30.53	31.8	38.2	**23.9**
	24	Age	Age	Age	Age
		30.68	**22.9**	32.1	24.9
	36	Age	Age	Age	Age
		19.68	22.3	20.4	**17.8**

5 Experimental Results

The documents of the examined corpus were downloaded from the Bar-Ilan University's Responsa Project[1]. The examined corpora contain 24,111 responsa written by 36 scholars, averaging 670 files for each scholar. These authors lived over a period of 250 years (1765–2015). These files contain citations; each citation pattern can be expanded into many other specific citations [12].

The citation recognition in this research is done by comparing each word to a list of 339 known authors and many of their books. This list of 25,801 specific citations that relate to names, nick names and abbreviations of these authors and their writings. Basic citations were collected and all other citations were produced from them.

We divide the data into three sets of authors documents (1) 12 scholars: containing 10,561 files, 880 files on average for each scholar spread over 135 years (1880–2015); (2) 24 scholars: containing 15,495 files, 646 files on average for each scholar spread over 229 years (1786–2015) (the set of 24 authors contains the set of 12 authors); and (3) 36 scholars: containing 24,111 files, 670 files on average for each scholar spread over 250 years (1765–2015) (the set of 36 authors contains the set of 24 authors). The research question (prediction of birth and death years of authors based on undated citations) that we face is relatively novel.

[1] The Global Jewish Database (The Responsa Project at Bar-Ilan University). Http://www.biu.ac.il/ICJI/Responsa.

Thus, there is no basis for comparison to assess the accuracy of the results. Because of this we use the distance function, i.e., we will measure the distance between the real birth and death years of the authors and the assessments of the algorithms in order to evaluate the results.

Because we have three groups of authors; each one with a different span of years, as mentioned before, we have to normalize the results in order to compare between them. The results of the original 12 authors are multiplied by 1.85 when compared to the results of 36 authors (the amplitude years of 12 authors is 135, the amplitude years of 36 authors is 250, result 250 / 135 ~ 1.85), we did the same for the set 24 authors (the amplitude years of 24 authors is 229, the amplitude years of 36 authors is 250, result 250 / 229 ~ 1.09).

The results that appear in the following tables, each table shows results of two algorithms - Iron+Heuristic (sub-section 3.1) and Greedy (sub-section 3.2). Each algorithm was performed on three groups of authors: 12 authors, 24 authors, and 36 authors. For both algorithm executions there are results containing estimated years of birth and death. The results shown in the tables are the best birth/death date deviation results. In every quarter table there are four columns: a deviation without refinement, a deviation with the "Late" refinement, with the "Rabbi" refinement, and with the "Friend" refinement (Section 3).

In addition, we used two manipulations - Age and Current year (sub-section 3.3). The bold cells contain the best results. The following four tables contain the results of the evaluations of birth and death years of the two algorithms, i.e., the Greedy algorithm and the Iron+Heuristic algorithm, with the use of the years-feature (see (3 (H)) and (10(G))), total of 96 results.

The Age manipulation gives the best results with 76% for all refinements, in both algorithms, with or without constants, i.e. in the two tables (73/96=0.76). It doesn't necessarily mean that to all the authors Age manipulation was done, but for some of them it was necessary.

This manipulation is effective, because it is only used when estimate is erratic; therefore, a manipulation is necessary and usually it improves the results. For example, in the Greedy algorithm, when using years with the "Rabbi" refinement for 36 authors, the estimated birth years and death years are improved in 44 results out of 72 (29 for birth year estimates and 15 for death year estimates); the average improvement estimate was 15.9 years for all 72 results (36 birth years and 36 death years).

In the Iron+Heuristic algorithm, when using the years and "friend" refinements for 36 authors, the estimated birth years and death years are improved in 45 results out of 72 (22 for birth year estimates and 23 for death year estimates); the average improvement estimate was 11.13 years for all 72 results (36 birth years and 36 death years). To summarize, the Age manipulation is very helpful for the birth and death year reckoning.

First, we will analyze the algorithms from a general perspective, consistency; there are two aspects in terms of consistency: (1) at the level of the best results of the Iron+ Heuristic/Greedy algorithms for birth/death year assessments and (2) at the level of a specific refinement. (1) Are the best results of the 36 authors are better than the best results of the 24 authors and also are the best results of the 24 authors are better than the best results of the 12 authors? (2) For each specific refinement, is the result of the

Table 3. Birth average distance.

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	Age	const	Age	Age
		63.52	58.28	60.78	**26.32**
	24	Age	const	Age	Age
		42.96	31.72	33.64	**27.47**
	36	Age	const	Age	Age
		38.67	26.1	30.89	**21.18**
Greedy	12	Age	const	const	Age
		23.67	24.36	**22.82**	28.67
	24	const	Age	const	Age
		12.67	**12.38**	15.44	16.74
	36	const	Age	Age	Age
		11.63	**11.35**	14.17	15.35

with years-feature.

Table 4. Death average distance.

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	Age	no tuning	const	Age
		18.35	22.5	34.3	28.3
	24	no tuning	no tuning	Age	Age
		14.4	20.1	33.1	25.3
	36	no tuning	no tuning	Age	Age
		12.86	18.2	20.2	17.7
Greedy	12	Age	Age	Age	Age
		19.89	32.6	35.8	23.9
	24	Age	Age	Age	Age
		25.55	**22.9**	31.2	27.2
	36	Age	Age	Age	Age
		23.44	**21**	28.7	25

with years-feature.

36 authors better than the result of the 24 authors and for the same refinement, is the result of the 24 authors is better than result of the 12 authors.

In the Iron+Heuristic algorithm there is an almost complete consistency in the two aspects, namely the more authors and more information the deviation results become better. For example, from the first perspective, in table 2 the best result of 36 authors is 17.72, the best result of 24 authors is 19.92, and the best result of 12 authors is 28.33; the more authors the better the predictions. Example for the second perspective, in table 2 with the Rabbi refinement, the best result of 36 authors is 20.03, the best result of 24 authors is 32.73, and the best result of 12 authors is 45.13.

There is one case of inconsistency at the Iron+Heuristic algorithm in table 3 with the friend refinement, where the result became worse but with a relatively small deviation (about 1.1). However, compared with the Iron+Heuristic algorithm, in the Greedy algorithm there is greater inconsistency in both levels. For instance, from the first aspect, table 4 shows that the best result of 36 authors is 20.98, the best result of the 24 authors is 22.87, and the best result of 12 authors is 19.8; from the second aspect, reviewing table 4 in the Rabbi refinement, the best result of 36 authors is 24.98, the best result of 24 authors is 27.23, and the best result of 12 authors is 23.9.

A possible explanation is that the Greedy algorithm is an intuitive therefore its consistency is "not his forte"; In contrast, the Iron+Heuristic algorithm is mathematically "committed" (up to the heuristics) and therefore it is much more consistent. In conclusion, the more information the Iron+Heuristic algorithm is more stable and more consistent.

Table 5. Birth average distance without years, Mughaz et al. [13].

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	49.29	93.43	51.15	23.79
	24	37.68	44.8	35.27	24.94
Greedy	12	31.01	54.33	39.38	33.09
	24	24.45	30.25	26.74	22.32

Table 6. Death average distance without years, Mughaz et al. [13].

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	45.25	31.19	45.57	33.87
	24	26.4	16.54	28.53	23.14
Greedy	12	29.91	32.01	38.28	23.03
	24	30.78	22.91	32.15	26.76

Table 7. Birth average distance without years, Current results.

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	46.56	60.97	37.5	26.32
	24	34.93	33.65	28.58	26.13
Greedy	12	30.22	39.85	25.28	28.37
	24	15.28	19.98	17.46	15.71

Table 8. Death average distance without years, Current results.

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	43.1	41.47	45.13	28.33
	24	23.98	19.92	32.73	23.98
Greedy	12	30.53	31.84	38.16	23.9
	24	30.68	22.87	32.09	24.89

When we use the Iron+Heuristic algorithm in order to evaluate the birth years we can see from tables 1 and 3 that unequivocally that the friend refinement always gives the best results compared to the other refinements whether we use the years-feature or whether we do not. It is arises from formula (7(H)), that the output of the core of formula (7(H)), i.e., MIN(B(Yi)), is much closer to the real birth year so that the use of a constant in (7(H)), i.e., (MAX-MIN), will lead to better results relative to the other refinements.

For example, in formula (7(H)), (which serves the Rabbi refinement and the friend refinement) the MIN(B(Yi)) of the Rabbi refinement is less than the MIN (B(Yi)) of the friend refinement (usually the Rabbi of X born before the friend of X), therefore, the average results of the Rabbi refinement will not be as good as the results of the friend refinement; similarly in the "late" refinement in formula (6(H)). As opposed to the Iron+Heuristic algorithm, the Greedy algorithm does not have consistency.

In the Greedy algorithm, the best result of the set of 12 authors was obtained using the friend refinement; the best results of sets of 24 and 36 authors, were obtained without the use of years-feature are using “no refinement” and with the use of years-feature are using the "late" refinement.

Although the Iron+Heuristic algorithm is more stable and consistent, the results of the Greedy algorithm are better (both with and without the use of the years-feature). When we analyze the differences between the numerical results of the two algorithms, we see that the results of the Greedy algorithm are better in all cases except two cases

Table 9. Birth average distance with years, Mughaz et al. [13].

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	94.25	93.5	74.55	23.72
	24	59.04	44.8	45.24	26.79
Greedy	12	95.79	54.33	76.8	33.09
	24	54.36	30.25	42.67	25.85

Table 10. Death average distance with years, Mughaz et al. [13].

	No refinement	Late	Rabbi	Friend
Iron + Heuristic	17.99	31.27	32.23	33.95
	14.39	16.54	25.99	25
Greedy	19.53	32.33	35.41	23.03
	25.62	22.91	31.31	28.51

Table 11. Birth average distance without years, Current results.

	# of authors	No refinement	Late	Rabbi	Friend
Iron + Heuristic	12	63.52	58.28	60.78	26.32
	24	42.96	31.72	33.64	27.47
Greedy	12	23.67	24.36	22.82	28.67
	24	12.67	12.38	15.44	16.74

Table 12. Death average distance without years, Current results.

	No refinement	Late	Rabbi	Friend
Iron + Heuristic	18.35	22.51	34.3	28.33
	14.4	20.05	33.1	25.3
Greedy	19.89	32.61	35.77	23.9
	25.55	22.87	31.22	27.23

of 12 authors with the friend refinement (a difference average of 2.2 years, for the two cases).

The average years' deviation without the use of years-feature for the Greedy algorithm is 23.7 and for the Iron+Heuristic algorithm is 30.34. The average years' deviation with the use of years-feature for the Greedy algorithm is 21.93 and for the Iron+Heuristic algorithm is 30.28.

When we are considering only the best results for each set of authors, without the use of the years-feature in the Iron+Heuristic algorithm the average years' deviation is 23 and in the Greedy algorithm the average years' deviation is 19.59; i.e., the results of the Greedy algorithm are more accurate. When we use the years-feature, the years' deviations of the Iron+Heuristic and the Greedy algorithms are 20.1 and 18.38, respectively. In conclusion, the numerical results of the Greedy algorithm are better but the results of the Iron+Heuristic algorithm are more stable and consistent.

Current research versus previous research

In this research, various additions are presented comparing to Mughaz et al. [13] (We are competing with [13] and not with 12 or 11 because 13 is the advanced among them):

1 There are three corpora of responsa composed by 12, 24 and 36 authors, instead of tow corpora (12 and 24 authors),

2 There is a use of three different time intervals instead of only two (Certainly we normalize the results accordingly),

3 We analyzed the stability and consistency of the two algorithms,

4 We checked and used different values of constants where at Mughaz et al. [13] they use the same values of constants,

5 We extend the formula of "years-feature", i.e., (3 (H)) and (10(G)).

Mughaz et al. [13] examined a corpus, which includes 15,450 responsa where 10,512 responsa of them were written by 12 scholars and 15,450 responsa were written by 24 scholars. In this research, the corpus contains 24,111 responsa where 10,561 responsa of them were written by 12, 15,495 responsa were written by 24 scholars, and 24,111 responsa written by 36 scholars. The 15,450 responsa used in Mughaz et al. [13] are included this research.

When we consider the results of Mughaz et al. [13] study versus the results of the current study we see that we have improved 44 results out of 64 results (of 12 and 24 authors), i.e., an improvement of 69%. In more details: The Iron+Heuristic algorithm presents better results of 5.49 years on average (for all the 36 experiments); with using the years-feature the results are better in 7.39 years on average, without using the years-feature the results are better in 3.6 years on average. In 22 cases out of 36 the results are better and in 14 cases are worse.

The Greedy algorithm presents better results of 10.2 years on average (for all the 36 experiments); with using the years-feature the results are better in 16 years on average, without using the years-feature the results are better in 4.39 years on average. In 30 cases out of 36 the results are better and in 6 cases are worse.

6 Summary, Conclusions and Future Work

We investigate the estimation of the birth and death years of the authors using undated citations referring to them or written by them. This research was performed on a special case of documents (i.e., responsa), where special writing rules are applied. The estimation was based on the author's documents and documents of other authors who refer to the discussed author or are mentioned by him. To do so, we use various kinds of iron-clad, heuristic and greedy constraints.

In this research we show an improvement of 44 results out of 64 results, i.e., an improvement of 69%.The examination of the estimation of the birth and death year indicate that the Greedy algorithm has been obtain better results than of the Iron+Heuristic algorithm but the stability and consistency Iron+Heuristic algorithm is better.

Regarding the estimation of the birth and death years of an author X, it is important to point that citations mentioned by X or referring to X are more suitable to assess the "birth" and "death" writing years of X rather than his real birth and death years.

This model can be applied with suitable changes to similar research problems that might be relevant for some historical document collections.

We plan to improve the assessment of the birth and death years of authors by: (1) Combining and testing new combinations of iron-clad, heuristic and greedy constraints, (2) Improving existing constraints and/or formulating new constraints, (3) Defining and applying heuristic constraints that take into account various details included in the responsa, e.g., events, names of people, concepts, special words and collocations that

can be dated, (4) Conducting additional experiments using many more responsa written by more authors is supposed to improve the estimates, (5) Checking why the iron-clad, heuristic and greedy constraints tend to produce more positive differences, and (6) Testing how much of an improvement we got from a correction of the upper bound of D(x) and how much we will at some point use it for a corpus with long-dead authors.

References

1. HaCohen-Kerner, Y., Schweitzer, N., Mughaz, D.: Automatically identifying citations in hebrew-aramaic documents. Cybernetics and Systems: An International Journal, vol. 42 no. 3, pp. 180–197 (2011) doi: 10.1080/01969722.2011.567893
2. Wintner, S.: Hebrew computational linguistics: Past and future. Artificial Intelligence Review. vol. 21, no. 2, pp. 113–138 (2004) doi:10.1023/B:AIRE.0000020865.73561.bc
3. HaCohen-Kerner, Y., Kass A., Peretz, A.: Baseline methods for automatic disambiguation of abbreviations in Jewish law documents. In: Proceedings of the 4th International Conference on Advances in Natural Language, pp. 58–69 (2004) doi:10.1007/978-3-540-30228-5_6
4. HaCohen-Kerner, Y., Kass A., Peretz, A.: Abbreviation disambiguation: Experiments with various variants of the one sense per discourse hypothesis. In: Proceedings of the Application of Natural Language to Information Systems (NLDB'08), pages 27–39 (2008) doi: 10.1007/978-3-540-69858-6_5
5. HaCohen-Kerner, Y., Kass A., Peretz, A.: Combined one sense disambiguation of abbreviations. ACL, pp. 61–64 (2008)
6. HaCohen-Kerner, Y., Kass A., Peretz, A.: HAADS: A hebrew aramaic abbreviation disambiguation system. Journal of the American Society for Information Science and Technology, vol. 61, no. 9, pp. 1923–1932 (2010)
7. HaCohen-Kerner, Y., Kass A., Peretz, A.: Initialism disambiguation: Man versus machine. Journal of the American Society for Information Science and Technology, vol. 64, no. 10, pp. 2133–2148 (2013) doi: 10.1002/asi.22909
8. Mughaz, D., HaCohen-Kerner, Y., Gabbay, D.: Estimating the birth and death years of authors of undated documents using undated citations. In: International Conference on Natural Language Processing. Springer, pp. 138–149 (2010) doi: 10.1007/978-3-642-14770- 8_17
9. Mughaz, D., HaCohen-Kerner, Y., Gabbay, D.: When text authors lived using undated citations. In: Information Retrieval Facility Conference, Springer, vol. 8849, pp. 82–95 (2014) doi: 10.1007/978-3-319-12979-2_8
10. Mughaz, D., HaCohen-Kerner, Y., Gabbay, D.: Key-Phrases as means to estimate birth and death years of jewish text authors. Semanitic Keyword-based Search on Structured Data Sources, Springer, pp. 108–126 (2015) doi: 10.1007/978-3-319-27932-9_10
11. Garfield, E.: Can citation indexing be automated? Statistical Association Methods for Mechanical Documentation, In: Symposium Proceedings, National Bureau of Standards Miscellaneous Publication, vol. 269, pp. 189–142 (1965)
12. Berkowitz, E., Elkhadiri, M. R.: Creation of a style independent intelligent autonomous citation indexer to support academic research. pp. 68–73 (2004)
13. Giuffrida, G., Shek, E. C., Yang, J.: Knowledge-based metadata extraction from postscript files. In: Proceedings of the 5th ACM conference on Digital libraries, ACM, pp. 77–84 (2000) doi:10.1145/336597.33663
14. Seymore, K., McCallum, A., Rosenfeld, R.: Learning hidden Markov model structure for information extraction. In: Workshop on Machine Learning for Information Extraction, pp. 37–42 (1999)

15. Bradshaw, S.: Reference directed indexing: redeeming relevance for subject search in citation indexes. Research and advanced technology for digital libraries, Springer, pp. 499–510 (2003)
16. Ritchie, A., Teufel, S., Robertson, S.: Using terms from citations for IR: some first results. In: The European Conference for Information Retrieval (ECIR), pp. 211–221 (2007) doi: 10.1007/978-3-540-78646-7_21
17. Ritchie, A., Robertson, S., Teufel, S.: Comparing citation contexts for information retrieval. In: 17th ACM Conference on Information and Knowledge Management (CIKM), pp. 213–222 (2008) doi: 10.1145/1458082.145811
18. Teufel, S., Siddharthan, A., Tidhar, D.: Automatic classification of citation function. In: Conference on Empirical Methods in Natural Language Processing, pp. 103–110 (2006)
19. Kolomiyets, O., Bethard, S., Moens, M. F.: Extracting narrative timelines as temporal dependency structures. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pp. 88–97 (2012)
20. Bethard, S., Kolomiyets, O., Moens, M. F.: Annotating story timelines as temporal dependency structures. In: Proceedings of the eighth international conference on language resources and evaluation, ELRA, pp. 2721–2726 (2012)
21. Schwartz, H. A., Park, G. J., Sap, M., Weingarten, E., Eichstaedt, J. C., Kern, M. L., Ungar, L. H.: Extracting human temporal orientation from facebook language. HLT-NAACL, pp. 409–419 (2015)
22. Wen, M., Zheng, Z., Jang, H., Xiang, G., Rosé, C. P.: Extracting events with informal temporal references in personal histories in online communities. ACL, vol. 2, pp. 836–842 (2013)
23. Kim, S. N., Medelyan, O., Kan, M. Y., Baldwin, T.: Automatic key-phrase extraction from scientific articles. Language resources and evaluation, vol. 47, no. 3, pp. 723–742 (2013)
24. Yih, W. T., Goodman, J., Carvalho, V. R.: Finding advertising key-words on web pages. In: Proceedings of the 15th international conference on World Wide Web, ACM, pp. 213–222 (2006)
25. Mughaz, D.: Classification of hebrew texts according to style. Thesis (in Hebrew), Bar-Ilan University (2003)
26. Koppel, M., Mughaz, D., Akiva, N.: CHAT: A system for stylistic classification of hebrew-aramaic texts. In: The 3th Workshop on Operational Text Classification Systems (OTC-03), vol. 27 (2003)
27. Koppel, M., Mughaz, D., Akiva, N.: New methods for attribution of rabbinic literature. Hebrew Linguistics, A Journal for Hebrew Descriptive, Computational, Applied Linguistics, University Press, vol. 57 (2006)
28. Koppel, M., Mughaz, D., Schler, J.: Text categorization for authorship verification. In: 8th International Symposium on Artificial Intelligence and Mathematics (2004)
29. Liebeskind, C., Dagan, I., Schler, J.: Statistical thesaurus construction for a morphologically rich language. In: Proceedings of the First Joint Conference on Lexical and Computational Semantics, Proceedings of the main conference and the shared task, Proceedings of the Sixth International Workshop on Semantic Evaluation, pp. 59–64 (2012)
30. Liebeskind, C., Dagan, I., Schler, J.: Semi-automatic construction of cross-period thesaurus. LaTeCH'13, vol. 29 (2013)
31. Liebeskind, C., Dagan, I., Schler, J.: Semiautomatic construction of cross-period thesaurus. Journal on Computing and Cultural Heritage (JOCCH), vol. 9, no. 4, pp. 22 (2016)
32. Boyack, K. W., Small, H., Klavans, R.: Improving the accuracy of co-citation clustering using full text. Journal of the American Society for Information Science and Technology (JASIST), vol. 64, no. 9, pp. 1759–1767 (2013)
33. Athar, A., Teufel, S.: Context-enhanced citation sentiment detection. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational

Linguistics: Human Language Technologies Association for Computational Linguistics, ACL, pp. 597–601 (2012)
34. Powley, B., Dale, R.: Evidence-based information extraction for high accuracy citation and author name identification. RIAO'07, pp. 618–632 (2007)
35. Ritchie, A., Teufel, S., Robertson, S.: Using terms from citations for IR: some first results. In: European Conference for Information Retrieval (ECIR), pp. 211–221 (2007)

ISSN 1870-4069

Reasoning with Self-attention and Inference Model for Machine Comprehension

Zhuang Liu, Degen Huang,
Kaiyu Huang, Jing Zhang

Dalian University of Technology,
School of Computer Science and Technology,
China

{zhuangliu, zhangjingqf, huangkaiyu}@mail.dlut.edu.cn
huangdg@dlut.edu.cn

Resumen. Enabling a computer to understand a document so that it can answer comprehension questions is a central, yet unsolved goal of Natural Language Processing (NLP), so reading comprehension of text is an important problem in NLP. Recently, machine reading comprehension has embraced a booming in NLP research. In this paper, we introduce a novel iterative inference neural network based on a matrix sentence embedding with a self-attention mechanism. The proposed approach continually refines its view of the query and document while aggregating the information required to answer a query, aiming to compute the attentions not only for the document but also the query side, which will benefit from the mutual information. Experimental results show that our model has achieved significant state-of-the-art performance in public English datasets, such as CNN and Children's Book Test datasets. Furthermore, the proposed model also outperforms state-of-the-art systems by a large margin in Chinese datasets, including People Daily and Children's Fairy Tale datasets, which are recently released and the first Chinese reading comprehension datasets.

Palabras clave: Machine comprehension, matrix sentence, hybrid models of reading comprehension.

1 Introduction

Reading comprehension[1] is the ability to read text, process it, and understand its meaning. How to endow computers with this capacity has been an elusive challenge and a long-standing goal of Artificial Intelligence. A recent trend to measure progress towards machine reading is to test a system's ability to answering questions over the text it has to comprehend.

Towards this end, several large-scale datasets of Cloze-style questions over a context document have been introduced recently which allow the training of supervised machine learning systems [4, 6, 7, 23]. Cloze-style queries are representative problems in reading comprehension. Over the past few months, we have seen much progress that is utilizing neural network approach to solve Cloze-style questions.

[1] en.wikipedia.org/wiki/Reading_comprehension

In the past year, to teach the machine to do Cloze-style reading comprehensions, large-scale training data is necessary for learning relationships between the given document and query. Some large-scale reading comprehensions datasets have been released: the CNN/Daily Mail corpus, consisting of news articles from those outlets [6], and the Children's Book Test (CBTest), consisting of short excerpts from books available through Project Gutenberg [7].

Recently, Cui et al. [4] has released the first machine Chinese reading comprehension datasets, including a human-made out-of-domain test set for future research. All previous works are focusing on automatically generating large-scale training data for neural network training, which demonstrate its importance. Furthermore, the more complicated problems the more data is needed to learn comprehensive knowledge from it, such as reasoning over multiple sentences etc.

In this paper, we propose a novel iterative inference neural network model, designed to study machine comprehension of text, which constructs iterative inference mechanism. The Self-Attentive Encoder Model first uses a self-attention mechanism for representing sentences.

Then, the core module, Iterative Inference Model, begins by deploying an iterative inference mechanism that alternates between attending query encodings and document encodings, to uncover the inferential links that exist between the document and the query. The results of the alternating attention are gated and fed back into the inference LSTM. After a number of steps, the weights of the document attention are used to estimate the probability of the answer.

To sum up, our contributions can be summarized as follows:

- We propose a novel end-to-end neural network model for machine reading comprehension, which combines self-attentive sentence embedding and an iterative inference mechanism to handle the Cloze-style reading comprehension task.
- Also, we have achieved the state-of-the-art performance in public reading comprehension datasets, including English datasets and Chinese datasets.
- Our further analyses with the models reveal some useful insights for further improving the method.

2 Problem Notation, Datasets

2.1 Definition and Notation

The task of the proposed model is to answer a Cloze-style question by reading and comprehending a supporting passage of text. The Cloze-style reading comprehension problem (proposed by Taylor[20]) aims to comprehend the given context or document, and then answer the questions based on the nature of the document, while the answer is a single word in the document. Thus, the Cloze-style reading comprehension can be described as a triple:

$$(Q, D, A), \tag{1}$$

where Q is the query (represented as a sequence of words), D is the document, A is the set of possible answers to the query.

Document	1 ||| 人民网 1月 1日 讯 据 《 纽约 时报 》 报道 ， 美国 华尔街 股市 在 2013年 的 最后 一 天 继续 上涨 ， 和 全球 股市 一样 ， 都 以 最高 纪录 或 接近 最高 纪录 结束 本年 的 交易 。 2 ||| 《 纽约 时报 》 报道 说 ， 标普 500 指数 今年 上升 29.6% ， 为 1997年 以来 的 最 大 涨幅 ； 3 ||| 道琼斯 工业 平均 指数 上升 26.5% ， 为 1996年 以来 的 最 大 涨幅 ； 4 ||| 纳斯达克 上涨 38.3% 。 5 ||| 就 12月 31日 来说 ， 由于 就业 前景 看好 和 经济 增长 明年 可能 加速 ， 消费者 信心 上升 。 6 ||| 工商 协进会 报告 ， 12月 消费者 信心 上升 到 78.1 ， 明显 高于 11月 的 72 。 7 ||| 另 据 《 华尔街 日报 》 报道 ， 2013年 是 1995年 以来 美国 股市 表现 最 好 的 一 年 。 8 ||| 这 一 年 里 ， 投资 美国 股市 的 明智 做法 是 追 着 " 傻钱 " 跑 。 9 ||| 所谓 的 " 傻钱 " **X** ， 其实 就 是 买 入 并 持有 美国 股票 这样 的 普通 组合 。 10 ||| 这个 策略 要 比 对冲 基金 和 其它 专业 投资者 使用 的 更为 复杂 的 投资 方法 效果 好 得 多 。
Query	所谓 的 " 傻钱 " **X** ， 其实 就 是 买 入 并 持有 美国 股票 这样 的 普通 组合 。
Answer	策略

Fig. 1. Example training sample in people daily datasets. The "**X**" represents the missing word. In this example, the document consists of 10 sentences, and the 9th sentence is chosen as the query.

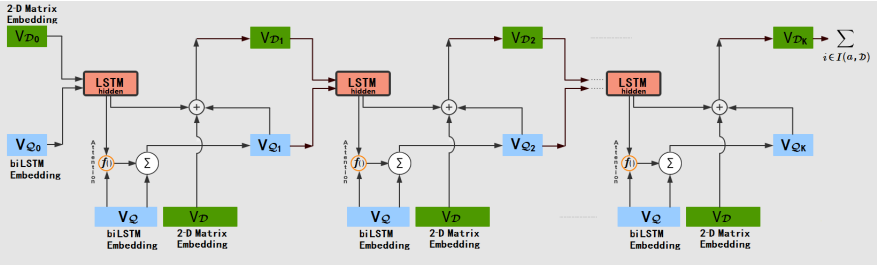


Fig. 2. Architecture of the proposed iterative inference neural networks.

2.2 Reading Comprehension Datasets

Several institutes have released the Cloze-style reading comprehension data, and these have greatly accelerated the research of machine reading comprehension. We begin with a brief introduction of the existing Cloze-style reading comprehension datasets, two English datasets and the first Chinese reading comprehension datasets recently released.

Typically, there are two main genres of the English Cloze-style datasets publicly available, CNN/Daily Mail [6] and Children's Book Test (CBTest) [7], which all stem from the English reading materials. Also, there are the first Chinese Cloze-style reading comprehension datasets, People Daily[4] and Children's Fairy Tale (CFT) [4], which are roughly collected 60K news articles from the People Daily website[2] and the Children's Fairy Tale by Cui et al. [4]. Figure 1 shows an example of People Daily datasets[3,4].

Table 1 provides some statistics on the two English datasets: CNN/Daily Mail and Children's Book Test (CBTest). The statistics of People Daily datasets as well as Children's Fairy Tale datasets[5] are listed in the Table 2.

3 Proposed Approach

In this section, we will introduce our iterative inference neural networks for Cloze-style reading comprehension task. The proposed model is shown in Figure 2.

[2]People's Daily: http://www.people.com.cn

[3]CNN and Daily Mail datasets are available at http://cs.nyu.edu/%7ekcho/DMQA

[4]CBTest datasets is available at http://www.thespermwhale.com/jaseweston/babi/CBTest.tgz

[5]People Daily and CFT datasets are available at http://hfl.iflytek.com/chinese-rc

Table 1. Data statistics of the CNN datasets and Children's Book Test datasets (CBTest). CBTest CN stands for CBTest Common Nouns and CBTest NE stands for CBTest Named Entites. CBTest had a fixed number of 10 options for answering each question. Statistics provided with the CBTest dataset.

	CNN			**CBTest CN**			**CBTest NE**		
	Train	**Valid**	**Test**	**Train**	**Valid**	**Test**	**Train**	**Valid**	**Test**
# queries	380,298	3,924	3,198	879,450	2,000	2,500	108,719	2,000	2,500
Max# options	527	187	396	10	10	10	10	10	10
Avg# options	26.4	26.5	24.5	10	10	10	10	10	10
Avg# tokens	762	763	716	470	448	461	433	412	424
Vocab. size	118,497			53,185			53,063		

Table 2. Data statistics of people daily datasets and children's fairy tale datasets (CFT).

	People Daily			**Children's Fairy Tale**	
	Train	**Valid**	**Test**	**Test-auto**	**Test-human**
# queries	870,710	3,000	3,000	1,646	1,953
Max# tokens in docs	618	536	634	318	414
Max# tokens in query	502	153	265	83	92
Avg# tokens in docs	379	425	410	122	153
Avg# tokens in query	38	38	41	20	20
Vocabulary size		248,160			NA

In encoder module, we opt for a new model for extracting an interpretable sentence embedding by introducing self-attention [11]. Instead of using a vector, it uses a 2-D matrix to represent the embedding, with each row of the matrix attending on a different part of the sentence. So we propose to apply the matrix sentence embedding with a self-attention mechanism for the document representation.

In core module, iterative inference module, our model is primarily motivated by Sukhbaatar et al. [19], Kadlec et al. [9] and Chen et al. [1], which aim to directly estimate the answer from the document, instead of making a prediction over the full vocabularies. But we have noticed that by just concatenating the final representations of the query RNN states are not enough for representing the whole information of query.

So we propose to utilize the repeated, tight integration between query and document attention, which allows the model to explore dynamically which parts of the query are most important to predict the answer, and then to focus on the parts of the document that are most salient to the currently attended query components.

3.1 Self-Attentive Encoder

Inspired by Lin et al. [11] and Liu et al. [13], we opted for self-Attentive sentence embedding for representing document. The query encodings and document encodings are represented separately as query vector V_Q, document V_D. Let d denote the dimension of word embeddings, and S a document sentence consisting of a sequence of

n words $(w_1, ..., w_n)$ which can be represented by a dense column matrix $\mathbf{W}$, who have the shape $n \times d$. We use a bidirectional LSTM to process the individual sentence, and concatenate each $\overrightarrow{h_t}$ with $\overleftarrow{h_t}$ to obtain a hidden state h_t.

Let the hidden unit number for each unidirectional LSTM be k, and note all the n h_ts as H, which has the shape $n \times 2k$, following Lin et al. [11], we choose a linear combination of the n LSTM hidden vectors in H. Computing the linear combination uses the self-attention mechanism. The attention mechanism takes the whole LSTM hidden states H as input, and outputs a vector of weights $\mathbf{a}$:

$$\mathbf{a} = softmax(\mathbf{w_{s2}}\, tanh(W_{s1}H^T)), \tag{2}$$

where W_{s1} is a weight matrix and $\mathbf{w_{s2}}$ is a vector of parameters. Then we sum up the LSTM hidden states H according to the weight $\mathbf{a}$ to get a vector representation $\mathbf{m}$, then we multiple $\mathbf{m}$ that focus on different parts of the sentence to represent the overall semantics of the sentence.

When we want r different parts to be extracted from the sentence, we extend the $\mathbf{w_{s2}}$ into a matrix, note it as W_{s2}, and the resulting annotation vector $\mathbf{a}$ becomes annotation matrix A:

$$A = softmax(W_{s2}\, tanh(W_{s1}H^T)). \tag{3}$$

We can deem Equation(2) as a 2-layer MLP without bias, whose parameters are $\{W_{s1}, W_{s2}\}$. The embedding vector then becomes an $n \times 2k$ embedding matrix M. We compute the r weighted sums by multiplying the annotation matrix A and LSTM hidden states H, the resulting matrix is the sentence embedding:

$$M = AH. \tag{4}$$

We then use the resulting matrix to represent the embedding, with each row of the matrix attending on a different part of the individual sentences of document $V_{\mathcal{D}}$.

3.2 Iterative Inference Model

This phase aims to uncover a possible inference chain that starts at the query and the document and leads to the answer. Figure 2 illustrates iterative inference module. We use a bilinear term instead of a simple dot product [7, 14, 19] in order to compute the importance of each query term in the current time step *t*. This simple bilinear attention has been successfully used in Luong et al. [15]. We formulate a query glimpse $\mathbf{q}_t$ at time step *t* by:

$$q_{i,t} = \underset{i=1,\ldots,|\mathcal{Q}|}{softmax}\, \tilde{\mathbf{q}}_i^T \mathbf{W}_q \mathbf{s}_{t-1}, \tag{5}$$

$$\mathbf{q}_t = \sum_i q_{i,t} \tilde{\mathbf{q}}_i, \tag{6}$$

where $q_{i,t}$ are the query attention weights and $\tilde{\mathbf{q}}_i$ are the query encodings.

Table 3. Results on the CNN news, CBTest NE (named entity) and CN (common noun) datasets. The result that performs best is depicted in bold face.

	CNN News		CBTest NE		CBTest CN	
	Valid	Test	Valid	Test	Valid	Test
Impatient Reader (Hermann *et al.* [6])	61.8	63.8	-	-	-	-
MemNN (window + self-sup.) (Hill *et al.* [7])	63.4	66.8	70.4	66.6	64.2	63.0
AS Reader (Kadlec *et al.* [9])	68.6	69.5	73.8	68.6	68.8	63.4
Stanford AR (Chen *et al.* [1])	72.4	72.4	-	-	-	-
Iterative Attention (Sordoni *et al.* [18])	72.6	73.3	75.2	68.6	72.1	69.2
CAS Reader (avg mode)(Cui *et al.* [4])	68.2	70.0	74.2	69.2	68.2	65.7
GA Reader (Dhingra *et al.* [5])	73.0	73.8	74.9	69.0	69.0	63.9
EpiReader (Trischler *et al.* [21])	**73.4**	74.0	75.3	69.7	71.5	67.4
AoA Reader (Cui *et al.* [3])	73.1	74.4	**77.8**	72.0	72.2	69.4
Our proposed model	73.0	**74.5**	77.1	**72.1**	**72.4**	**69.4**

Table 4. Results on people daily datasets and children's fairy tale (CFT) datasets. The result that performs best is depicted in bold face. CAS Reader (marked with †) are the most recent works.

	People Daily		Children's Fairy Tale	
	Valid	Test	Test-auto	Test-human
AS Reader	64.1	67.2	40.9	33.1
CAS Reader (avg mode)	65.2	68.1	1.3	35.0
CAS Reader (sum mode)	64.7	66.8	3.0	34.7
CAS Reader (max mode)	63.3	65.4	8.3	32.0
Our proposed model	**66.6**	**69.8**	**45.0**	**37.0**

Our method extends the Attention Sum Reader [9], and performs multiple hops over the input. The alternating attention continues by probing the document given the current query glimpse $\mathbf{q}_t$.

The document attention weights are computed based on both the previous search state $t-1$ and the currently selected query glimpse $\mathbf{q}_t$:

$$d_{i,t} = \underset{i=1,\ldots,|\mathcal{D}|}{softmax}\, \tilde{\mathbf{d}}_i^T \mathbf{W}_d[\mathbf{q}_t, \mathbf{s}_{t-1}], \tag{7}$$

$$\mathbf{d}_t = \sum_i d_{i,t}\tilde{\mathbf{d}}_i, \tag{8}$$

where $\tilde{\mathbf{d}}_i$ are the query encodings, $d_{i,t}$ are the attention weights for each word in the document, and the document attention is also conditioned on $\mathbf{s}_{t-1}$, so it makes the model perform transitive reasoning on the document side. This use previously obtained document information to future attended locations, which is particularly important for natural language inference tasks[19].

In iterative inference module, the inference is modeled by an additional LSTM (proposed by Hochreiter and Schmidhuber[8]). The recurrent network iteratively performs an alternating search step to gather information that may be useful to predict the answer.

The module performs an attentive read on the query encodings, resulting in a query glimpse $\mathbf{q}_t$ at each time step, then gives the current query glimpse $\mathbf{q}_t$, it extracts a conditional document glimpse $\mathbf{d}_t$, representing the parts of the document that are relevant to the current query glimpse. Both attentive reads are conditioned on the previous hidden state of the inference LSTM $\mathbf{s}_{t-1}$, summarizing the information that has been gathered from the query and the document up to time t, making it easier to determine the degree of matching between them. The inference LSTM uses both glimpses to update its recurrent state and thus decides which information needs to be gathered to complete the inference process.

Finally, after a fixed number of time-steps K, the document attention weights obtained in the last search step $d_{i,K}$ are used to predict the probability of the answer. We aggregate the probabilities for tokens which appear multiple times in a document before selecting the maximum as the predicted answer:

$$P(a|Q,\ D) = \sum_{i \in I(a,D)} d_{i,K}, \tag{9}$$

where $I(a, D)$ is the set of positions where token a appears in the document D, we then use cross-entropy loss between the predicted probabilities and true answers for training.

4 Experiments

4.1 Experimental Setups

In this section we present our experimental setup for assessing the performance of our iterative inference neural networks. For the self-attentive encoder module, the biLSTM is 300 dimension in each direction, the attention MLP has 180 hidden units instead, and sentence embeddings of document has 20 rows (the r).

We use 300 dimensional GloVe embeddings [16] to represent words, projected down to 200 dimensions, a number determined via hyperparameter tuning. In hidden Layer, we initialized the LSTM units with random orthogonal matrices [17]. In order to minimize the hyper-parameter tuning, we used stochastic gradient descent with the ADAM update rule [10] and learning rate of 0.01 or 0.025, with an initial learning rate of 0.01.

Due to the time limitations, we only tested a few combinations of hyper-parameters, while we expect to have a full parameter tuning in the future. The results are reported with the best model, which is selected by the performance of validation set. The code in this paper has been implemented with Keras framework [2] and our source code also has be released.

4.2 Results

We compared the proposed model with several baselines as summarized below. To verify the effectiveness of our proposed model, we first tested our model on public English datasets. Our evaluation is carried out on CNN news datasets [6] and CBTest NE/CN datasets[7], and the statistics of these datasets are given in Table 3.

The results on Chinese reading comprehension datasets are listed in Table 4, as we can see that, the proposed model significantly outperforms the most recent state-of-the-art CAS Reader in all types of test set, with a maximum.

English Reading Comprehension Datasets. In CNN news datasets, our model is almost on par with the AoA Reader, but we failed to outperform EpiReader. Meantime, In CBTest NE, though there is a drop in the validation set with 0.6% declines, there is a boost in the test set with an absolute improvements over other models, which suggest our model is effective. In CBTest CN dataset, our model gives modest improvements over the state-of-the-art systems. When compared with AoA Reader, our model shows a similar result, with slight improvements on test set, which demonstrate that our model is more general and powerful than previous works.

Chinese Reading Comprehension Datasets. In People Daily and CFT datasets, our proposed model outperforms all the state-of-the-art systems by a large margin, where a 1.4%, 1.7%, 2.0% and 2.0% absolute accuracy improvements over the most recent state-of-the-art CAS Reader in the validation and test set respectively. This demonstrates that our model is powerful enough to compete with Chinese reading comprehension, to tackle the Cloze-style reading comprehension task.

So far, we have good results in machine reading comprehension, all higher than most baselines above, verifying that our proposed model is useful, suggesting that iterative inference neural networks performed better on relatively difficult reasoning questions.

5 Related Work

Neural attention models have been applied recently to machine learning and natural language processing problems. Cloze-style reading comprehension tasks have been widely investigated in recent studies. We will take a brief revisit to the previous works.

Hermann et al. [6] proposed a methodology for obtaining large quantities of (Q, D, A) triples through news articles and its summary. Along with the release of Cloze-style reading comprehension dataset, they also proposed an attention-based neural network to tackle the issues above. Experimental results showed that the proposed neural network is effective than traditional baselines. Hill et al. [7] released another dataset, which stems from the children's books.

Different from Hermann et al. work [6], the document and query are all generated from the raw story without any summary, which is much more general than previous work. To handle the reading comprehension task, they proposed a window-based memory network, and self-supervision heuristics is also applied to learn hard-attention. Kadlec et al. [9] proposed a simple model that directly pick the answer from the document, which is motivated by the Pointer Network [22]. A restriction of this model is that, the answer should be a single word and appear in the document.

Results on various public datasets showed that the proposed model is effective than previous works. Liu et al. [12] proposed to exploit these reading comprehension models into specific task. They first applied the reading comprehension model into Chinese zero pronoun resolution task with automatically generated large-scale pseudo training data. Trischler et al. [21] adopted a re-ranking strategy into the neural networks and used a joint-training method to optimize the neural network.

6 Conclusions

In this paper we presented the novel iterative inference neural network by introducing a fixed size, matrix sentence embedding with a self-attention mechanism, and showed it offered improved performance for machine comprehension tasks. Among the large, public Chinese and English datasets, our model could give significant improvements over various state-of-the-art baselines, especially for Chinese reading comprehension corpora, on which our model outperformed state-of-the-art systems by a large margin.

As future work, we need to consider how we can utilize these datasets to solve more complex machine reading comprehension tasks (with less annotated data), and we are going to investigate hybrid reading comprehension models to tackle the problems that rely on comprehensive induction of several sentences. We also plan to augment our framework with a more powerful model for natural language inference.

Acknowledgments. We thank the reviewers for their instructive feedback. This work is supported by National Natural Science Foundation of China (No.61672127).

References

1. Chen, D., Bolton, J., Manning, C.D.: A thorough examination of the CNN/daily mail reading comprehension task. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. vol. 1, pp. 2358–2367 (2016)
2. Chollet, F.: Building autoencoders in keras. The Keras Blog 14, 1–17 (2016)
3. Cui, Y., Chen, Z., Wei, S., Wang, S., Liu, T., Hu, G.: Attention-over-attention neural networks for reading comprehension. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. vol. 1, pp. 593–602 (2016)
4. Cui, Y., Liu, T., Chen, Z., Wang, S., Hu, G.: Consensus attention-based neural networks for chinese reading comprehension. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers (2016)
5. Dhingra, B., Liu, H., Cohen, W.W., Salakhutdinov, R.: Gated-attention readers for text comprehension 1, 1832–1846 (2016)
6. Hermann, K.M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., Blunsom, P.: Teaching machines to read and comprehend. In: Proceedings of the 28th International Conference on Neural Information Processing Systems. vol. 1, pp. 1693–1701 (2015)
7. Hill, F., Bordes, A., Chopra, S., Weston, J.: The goldilocks principle: Reading children's books with explicit memory representations. In: International Conference on Learning Representations (2015)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)

9. Kadlec, R., Schmid, M., Bajgar, O., Kleindienst, J.: Text understanding with the attention sum reader network. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. vol. 1, pp. 908–918 (2016)
10. Kingma, D., Ba, J.: A method for stochastic optimization. In: International Conference on Learning Representations (2015)
11. Lin, Z., Feng, M., Dos-Santos, C.N., Yu, M., Xiang, B., Zhou, B., Bengio, Y.: A structured self-attentive sentence embedding. In: International Conference on Learning Representations (2017)
12. Liu, T., Cui, Y., Yin, Q., Wang, S., Zhang, W., Hu, G.: Generating and exploiting large-scale pseudo training data for zero pronoun resolution. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. vol. 1, pp. 102–111 (2016)
13. Liu, Y., Liu, Z., Chua, T.S., Sun, M.: Topical word embeddings. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. vol. 29, pp. 2418–2424 (2015)
14. Liu, Z., Huang, D., Zhang, J., Huang, K.: Research on attention memory networks as a model for learning natural language inference. In: Proceedings of the Workshop on Structured Prediction for NLP. pp. 18–24 (2016)
15. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1412–1421 (2015)
16. Pennington, J., Socher, R., Manning, C.: GloVe: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). vol. 14, pp. 1532–1543 (2014)
17. Saxe, A.M., McClelland, J.L., Ganguli, S.: Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In: International Conference on Learning Representations (2013)
18. Sordoni, A., Bachman, P., Trischler, A., Bengio, Y.: Iterative alternating neural attention for machine reading (2016)
19. Sukhbaatar, S., Szlam, A., Weston, J., Fergus, R.: End-to-end memory networks. In: Advances in neural information processing systems. pp. 2440–2448 (2015)
20. Taylor, W.L.: Cloze procedure: A new tool for measuring readability. Journalism Bulletin 30(4), 415–433 (1953)
21. Trischler, A., Ye, Z., Yuan, X., Suleman, K.: Natural language comprehension with the epireader. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 128–137 (2016)
22. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: Advances in Neural Information Processing Systems (2015)
23. Zhuang-Liu, Degen-Huang, K.H., Zhang, J.: DIM reader: Dual interaction model for machine comprehension. In: International Symposium on Natural Language Processing Based on Naturally Annotated Big Data China National Conference on Chinese Computational Linguistics. pp. 387–397 (2017)

ISSN 1870-4069

Spanish Morphology Generation with Deep Learning

Carlos Escolano, Marta R. Costa-jussà

Universitat Politècnica de Catalunya,
TALP Research Center,
Barcelona

{carlos.escolano, marta.ruiz}@tsc.upc.edu

Abstract. Morphology generation is the natural language processing task of generating word inflection information. In this paper, we propose a new classification architecture based on deep learning to generate gender and number in Spanish from non-inflected words. This deep architecture uses a concatenation of embedding, convolutional and recurrent neural networks. We obtain improvements compared to other standard machine learning techniques. Accuracy of our proposed classifiers reaches over 98% for gender and over 93% for number.

Keywords: Morphology, deep learning, spanish.

1 Introduction

Morphology generation consists in generating the inflection information of a word. For example, given a lemmatized sentence like *El casa pequeño* in Spanish, a morphological generator would output *La casa pequeña*. Generally speaking, a morphological simplified text allows to reduce vocabulary specially in highly inflected languages. There are many natural language processing tasks and other related applications that can benefit from morphology generation to boost its performance. There have been many works in morphological generation and some of them are in the context of the application of machine translation.

Most of the works for this application translate to a morphologically simplified target, and then use some morphological generation technique to find the final output. To name a few, for example, [15] build maximum entropy markov models for inflection prediction of stems; [3, 10] use conditional random fields (CFR) to predict one or more morphological features; and [6] use Support Vector Machines (SVMs) to predict verb inflections.

A different but related task is the one of Part-of-Speech (PoS) tagging which aims at labelling words with its corresponding syntactic role. For this task, the form of the word is not simplified, so the form itself contains much more information than in the task of morphology generation. In this field the number of works is huge, but most related works to our own would be: [8] where authors train a model to predict each individual fragment of a PoS tag by means of machine learning algorithms; [4] where authors propose a deep learning architecture for English PoS tagging; and [11] where authors perform fine-grained PoS tagging with feedforward

Table 1. Examples of text representations.

Model	Text
Original	La casa de la playa
PoS & Lemma	DA0FS0[el] NCFS000[casa] SPS00[de] DA0FS00[el] NCFS000[playa]
Simplified Text	DA00[el] NC000[casa] SPS00[de] DA00[el] NC000[playa]
Simplified-Gender	DA0S0[el] NCS000[casa] SPS00[de] DA0S0[el] NCS000[playa]
Simplified-Number	DA0F0[el] NCF000[casa] SPS00[de] DA0FS[el] NCF000[playa]

and bidirectional recurrent architectures. In this paper, we focus on the task of generating morphological attributes for a particular task and we propose a deep learning architecture which concatenates embedding, convolutional and recurrent neural networks. This architecture is particularly tested on generating number and gender for the Spanish language.

The inputs to our system are lemmas and their corresponding fine-grained PoS tag where information of number and gender has been removed. Given the nature of our architecture, it could be further generalized to generating the fine-grained PoS tag from lemmas and for any language. The rest of the paper is organised as follows. Section 2 describes the morphology generation architecture. Section 3 details the experimental framework and section 4 reports accuracy of the classifier compared to other state-of-the-art techniques. Finally, section 5 highlights main achievements of this work.

2 Morphology Generation Architecture

This section reports which is the input data to our task, it describes which is the architecture proposed and it explains the motivation of this architecture.

2.1 Input Data

To train our system we start from a Spanish corpus and we use a morphological analyzer to remove the information of gender and number. Table 1 shows an example. Given our simplified text in gender and number, we propose to train two different models: one to retrieve gender and another to retrieve number. Each model decides among three different classes. Classes for gender classifier are masculine (M), femenine (F) and none (N); and classes for number classifier are singular (S), plural (P) and none (N).

2.2 Description

Inspired by previous Collobert's work [4], as features for our classifier we use windows of words as input. Each word is represented by a fixed size window of words in which the central element is the one to classify.

Figure 1 shows an example of windows of length 3. Note that in the example "de" does not have a window assigned because this word is invariant in gender and number. Following the example, our classifiers do not have to train all types of words. Some types of words, such as prepositions (*a, ante, cabo, bajo, de...*), do not have gender or

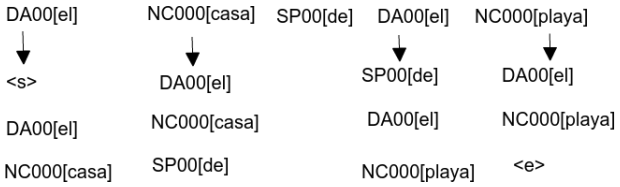


Fig. 1. Text to window representation.

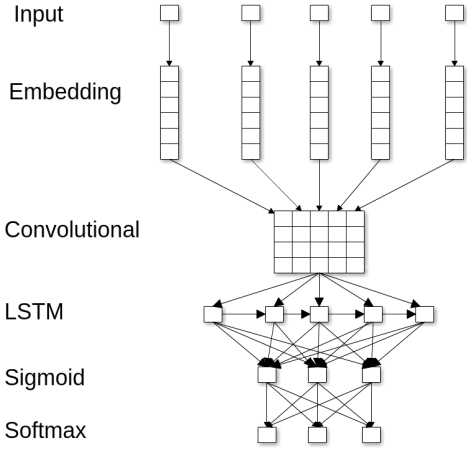


Fig. 2. Neural network overview.

number. Therefore our system was trained for determiners, adjectives, verbs, pronouns and nouns which are the ones that present morphology variations in gender or number. However, note that all types of words are used in the windows.

We base our architecture also in Collobert's proposal [4] and we modify it by adding a recurrent neural network. This recurrent neural network is relevant because it keeps information about previous elements in a sequence and, in our classification problem, context words are very relevant.

As a recurrent neural network, we use a Long Short Term Memory (LSTM) [9] that is proven to be efficient to deal with sequence NLP challenges [14]. This kind of recurrent neural network is able to maintain information for several elements in the sequence and to forget it when needed. Figure 2 shows an overview of the different layers involved in the final classification architecture, which are detailed as follows:

Embedding. We represent each word as its index in the vocabulary, i.e. every word is represented as one discrete value:

$$E(w) = W, W \in R^d. \tag{1}$$

w being the index of the word in the sorted vocabulary and d the size of the array. Then, each word is represented as a numeric array and each window is a matrix. This process is trained as part of the neural network architecture.

Table 2. Corpus details.

	Sentences	**Words**
Training	58,688	2,297,656
Development	990	43,489
Test	1010	44,306

Convolutional. We add a convolutional neural network. This step allows the system to detect some common patterns between the different words. This layer's input consists in a matrix W^l of multidimensional arrays of size $n \cdot d$, where n is the window length (in words) and d is the size of the array created by the previous embedding layer. This layer's output is a matrix of the same size as the input.

Max Pooling. This layer allows to extract most relevant features from the input data and reduces feature vectors to half.

LSTM. Each feature array is treated individually, generating a fixed size representation h_i of the ith word using information of all previous words (in the sequence). This layer's output, h, is the result of the last element of the sequence using information from all previous words.

Sigmoid. This layer smoothes results obtained by previous layer and compresses results to the interval $[-1, 1]$. This layer's input is a fixed size vector of shape $1 \cdot n$ where n is the number of neurons in the previous LSTM layer. This layer's output is a vector of length c equal to the number of classes to predict.

Softmax. This layer allows to show results as probabilities by ensuring that the returned value of each class belongs to the $[0, 1)$ interval and all classes add up 1.

2.3 Motivation

The input data of the classification algorithm is morphogically simplified in terms of gender and number. This simplification largely reduces the information that can be extracted from individual words in the vocabulary. In addition, we can encounter out-of-vocabulary words for which no morphological information can be extracted.

The main source of information are the context words. The information of a word consists in itself and the words that surround it (a window of words). Sometimes relevant information preceeds the word and sometimes information is after the word. Words (like adjetives), which are modifying or complementing another word, generally take information from preceeding words.

For example, in the sequence *casa blanca*, the word *blanca* could also be *blanco, blancos or blancas* but because noun and adjective are required to have gender and number agreement, the femenine word *casa* forces the femenine for *blanca*. While, for example, determiners usually take information from posterior words. This fact motivates that the word to classify has to be placed at the center of the window.

Finally, the fact that we rely only on the context information (since words themselves may not have any information) makes the recurrent neural network a key element in our architecture. The output h of the layer can be considered a context vector of the whole window maintaining information of the previous encountered words (in the same window).

Table 3. Distribution of the gender classes in the corpus.

Set	Word Type	Total	Femenine(%)	Masculine(%)	None(%)
Train	Determiners	340.739	53,24	38,36	8,40
	Nouns	571.053	52,19	46,52	1,29
	Verbs	219.638	14,24	12,75	73,01
	Pronouns	43.806	4,28	6,91	88,81
	Adjetives	185.107	21,69	24,14	54,17
Development	Determiners	6.534	51,18	41,20	7,62
	Nouns	11.025	51,39	46,78	1,83
	Verbs	5.630	11,10	13,20	75,70
	Pronouns	1.079	3,89	6,12	89,99
	Adjetives	3.129	60,95	38,69	0,36
Test	Determiners	6.858	52,07	40,76	7,17
	Nouns	11.347	51,72	46,56	1,72
	Verbs	4.629	12,18	13,23	74,59
	Pronouns	1.015	4,24	7,39	88,37
	Adjetives	3.375	24,68	24,21	51,11

3 Experimental Framework

This section reports the dataset used for experimentation together with its preprocessing. We also detail which are the parameters used for the final architecture configuration.

3.1 Data, Preprocessing and Software

As data, we use a subset of the United Nations Corpus [13]. Corpus statistics are shown in Table 2. Corpus preprocessing consisted in tokenization and lowercasing. PoS tagging was done using *Freeling* [12]. All chunking or name entity recognition was disabled to preserve the original number of words.

With the *Freeling* information, we represent each word with its PoS tag and lemma. From each PoS tag, we remove gender and number information to create the simplified text representation. For example, the word *La* becomes *DA0FS0[el]*, where *F* indicates its gender (femenine) and *S* its number (singular). This word is simplified to *DA00[el]*. A consequence of this new representation is that the determiners *el, la, los, las* are all represented as *DA00[el]* which introduces additional ambiguity to the task. See Table 1 for a full example of text simplification.

We do not consider for classification all word types. On the one hand, words that do not have explicit morphology according to *Freeling*'s tagset are not classified. However, these words are still used in the windows of words as context relevant information. On the other hand, word types being classified are *determiners, nouns, pronouns, adjectives, and verbs*, as shown in Tables 3 and 4. Balance details in gender and number for the different sets is detailed in Tables 3 and 4, respectively.

To generate the classification architecture we used the library *keras* [2] for creating and ensambling the different layers. Using NVIDIA GTX Titan X GPUs with 12GB of memory and 3072 CUDA Cores, each classifier has a training time of aproximately

Table 4. Distribution of the number classes in the corpus.

Set	Word Type	Total	Singular(%)	Plural(%)	None (%)
Train	Determiners	340.739	61,80	38,19	0,01
	Nouns	571.053	67,75	31,92	0,33
	Verbs	219.638	41,04	28,46	30,50
	Pronouns	43.806	12,21	8,09	79,70
	Adjetives	185.107	63,38	36,38	0,24
Development	Determiners	6.534	61,75	38,25	0
	Nouns	11.025	66,93	32,68	0,39
	Verbs	5.630	42,38	27,06	30,56
	Pronouns	1.079	9,55	7,04	83,41
	Adjetives	3.129	60,95	38,69	0,36
Test	Determiners	6.858	60,95	39,05	0
	Nouns	11.347	65,51	34,14	0,35
	Verbs	4.629	40,53	29,66	29,81
	Pronouns	1.015	9,66	9,16	81,18
	Adjetives	3.375	58,34	41,45	0,21

Table 5. Values of the different parameters of the classifiers.

Parameter	Gender	Number
Window size	7	9
Vocabulary size	7000	9000
Embedding	128	128
Filter size	5	7
LSTM nodes	70	70

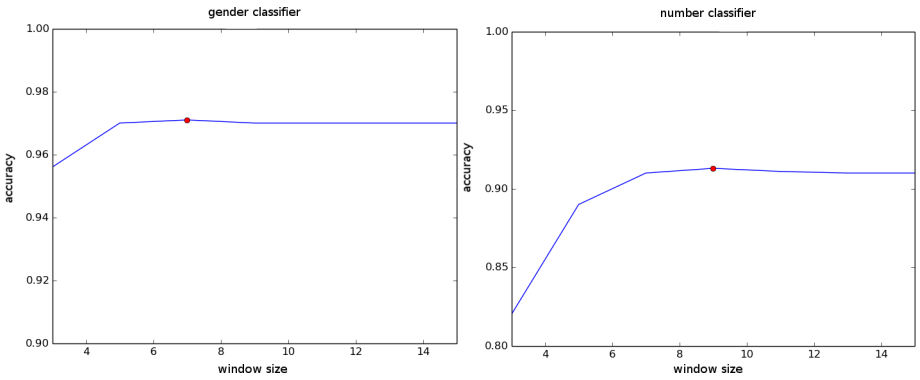


Fig. 3. Window size.

1h calculating two epochs of the model with batch size 32 and optimizer *adadelta* with default parameters.

3.2 Parameters Details

Regarding classification parameters, experimentation has shown that number and gender classification tasks have different requirements.

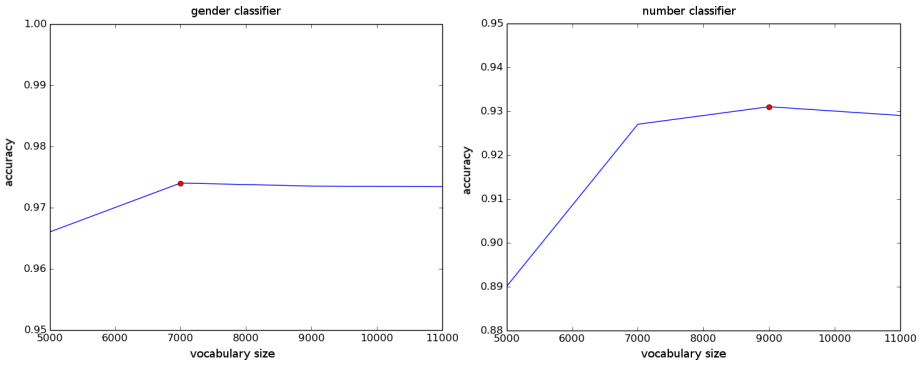


Fig. 4. Vocabulary size.

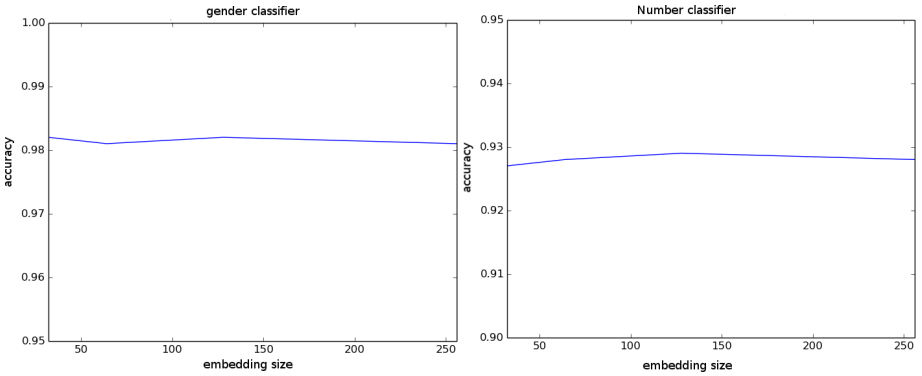


Fig. 5. Embedding size.

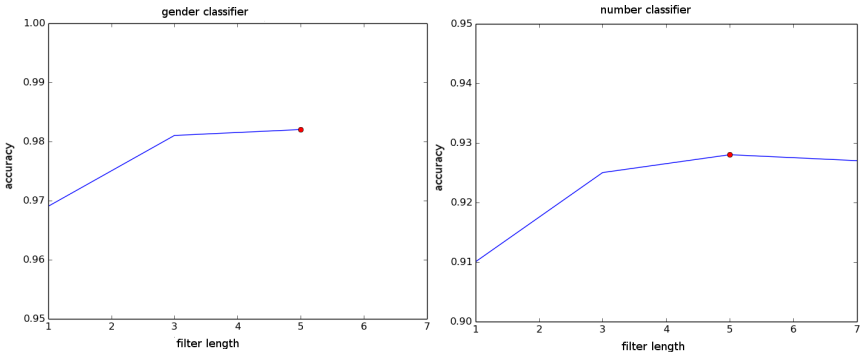


Fig. 6. Filter size.

Table 5 summarizes these parameters and details are given as follows.

– The best size of the window is found in 7 and 9 words for gender and number respectively. In both cases (number and gender) increasing this size lowers the accuracy of the system as shown in Figure 3.

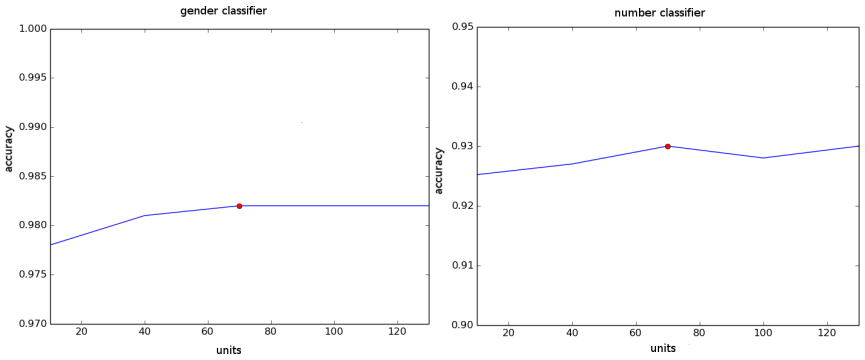


Fig. 7. LSTM nodes.

– The vocabulary size is a trade-off between giving enough information to the system to perform the classification while removing enough words to train the classifier for unknown words. We fix values to 7,000 and 9,000 for gender and number, respectively. The accuracy curve varying the vocabulary size is shown Figure 4).

– An embedding size of 128 results in stable training, while further increasing this value augmented the training time and hardware cost without impact in accuracy (see Figure 5). The filter size in the convolutional layer produced the best results when it was slightly smaller than the window size, being 5 and 7 the best values for gender and number classification, respectively (see Figure 6).

– Increasing LSTM nodes up to 70 improved significantly for both classifiers (see Figure 7).

4 Evaluation

Table 6 shows results for the classification task both number and gender. We have contrasted our proposed classification architecture based on neural networks with standard machine learning techniques such as linear, cuadratic and sigmoid kernels SVMs [5], random forests [1], convolutional[7] and LSTM[9] neural networks (NN). All algorithms were tested using features and parameters described in previous section with the exception of random forests in which we added the one hot encoding representation of the words to the features. We observe that our proposed architecture achieves by large the best results in all tasks.

5 Conclusions

This paper shows a new deep learning architecture for morphology generation. Our task is challenging because number and gender are generated from a word without this inflection. Our architecture uses several layers including embedding, convolutional and recurrent, which are able to outperform state-of-the-art techniques such as support vector machines or other well-known deep learning architectures.

Table 6. Classification results. In bold, best results.

Algorithm	**Accuracy**	
	Number	**Gender**
Naive Bayes	61.3	53.5
Lineal kernel SVM	68.1	71.7
Cuadratic kernel SVM	77.8	81.3
Sigmoid kernel SVM	83,1	87.4
Random Forest	81.6	91.8
Convolutional NN	81.3	93.9
LSTM NN	68.1	73.3
CNN + LSTM	**93.7**	**98.4**

We are able to improve accuracy almost by absolute 5% for gender classification and over 10% for number compared to convolutional neural networks and SVMs, respectively, which are the second-best performing systems. We reach over 98% accuracy for gender and over 93% accuracy for number. Further work includes using our architecture for PoS tagging and integrating it in a machine translation system.

Acknowledgments. This work is supported by the Spanish Ministerio de Economía y Competitividad and European Regional Development Fund, through the postdoctoral senior grant *Ramón y Cajal* and the contract TEC2015-69266-P (MINECO/FEDER, UE).

References

1. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
2. Chollet, F.: Keras: Deep learning for humans (2015), https://github.com/fchollet/keras
3. Clifton, A., Sarkar, A.: Combining morpheme-based machine translation with post-processing morpheme prediction. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 32–42 (2011)
4. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research 12, 2493–2537 (2011)
5. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
6. Formiga, L., Costa-jussà, M.R., Mariño, J.B., Fonollosa, J., Barrón-Cedeño, A., Màrquez, L.: The TALP-UPC phrase-based translation systems for WMT13: System combination with morphology generation, domain adaptation and corpus filtering. In: Proceedings of the Eighth Workshop on Statistical Machine Translation. pp. 134–140 (2013)
7. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics 36(4), 193–202 (1980)
8. Giménez, J., Màrquez, L.: SVMTool: A general POS tagger generator based on support vector machines. In: Proceedings of the 4th International Conference on Language Resources and Evaluation. pp. 43–46 (2004)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9(8), 1735–1780 (1997)

10. Kholy, A.E., Habash, N.: Rich morphology generation using statistical machine translation. In: Proceedings of the Seventh International Natural Language Generation Conference. pp. 90–94 (2012)
11. Labeau, M., Löser, K., Allauzen, A.: Non-lexical neural architecture for fine-grained POS tagging. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 232–237 (2015)
12. Padró, L., Stanilovsky, E.: Freeling 3.0: Towards wider multilinguality. In: Proceedings of the Language Resources and Evaluation Conference (LREC'12). pp. 2473–2479 (2012)
13. Rafalovitch, A., Dale, R.: United Nations general assembly resolutions: A six-language parallel corpus. In: Proceedings of Machine Translation Summit XII: Posters. pp. 292–299 (2009)
14. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Proceedings of the 27th International Conference on Neural Information Processing Systems. vol. 2, pp. 3104–3112 (2014)
15. Toutanova, K., Suzuki, H., Ruopp, A.: Applying morphology generation models to machine translation. In: Proceedings of the conference of the Association for Computational Linguistics and Human Language Technology (ACL-HLT). pp. 514–522 (2008)

ISSN 1870-4069

WikiAug: Augmenting Wikipedia Stubs by Suggesting Credible Hyperlinks

Ayesha Siddiqa, Ashish V. Tendulkar,
Sutanu Chakraborti

Indian Institute of Technology Madras,
India

{m.ayeshasiddiqa, ashishvt}@gmail.com
sutanuc@cse.iitm.ac.in

Abstract. Wikipedia is the ubiquitous resource for knowledge acquisition for humans and widely used systems like Apple's Siri, IBM's Watson and Google's Knowledge Graph. Thus, it is crucial that Wikipedia's articles are timely and accurate. Currently, Wikipedia articles are created and maintained by volunteers who may not be experts of the field. Consequently, many related concepts and relevant literature pointers may be missed by the editors. Furthermore, owing to the huge size and growth rate of Wikipedia, a manual search for information and maintenance process is unworkable in the long run. Our proposed approach *WikiAug* attempts to fill in these gaps and recommend concepts which are missing from Wikipedia articles (more specifically Stubs). To recommend concepts to Wikipedia articles, we rely on external knowledge retrieved by a search engine and semantic information like category labels and structure available in Wikipedia articles. This semantic knowledge adds new dimensions to the state-of-the-art approaches.

Keywords: Hyperlinks, link augmentation, wikipedia augmentation, explicit semantic analysis.

1 Introduction

Wikipedia has emerged as a reliable, comprehensive and authoritative encyclopedia on the Web. As of today, the English Wikipedia contains *5* million articles which contain rich semantic information in the form of text content, hyperlinks etc. Typically these links are links to other Wikipedia articles and external websites which contain related information. Carefully placed hyperlinks present users with new related concepts[1] and are essential for content discoverability. Widely used applications which rely on Wikipedia for their knowledge acquisition are Google's Knowledge Graph and Apple's Siri system. Knowledge acquired from Wikipedia is harvested and utilized in building knowledge bases like YAGO [19], DBpedia [4], and has interesting applications in Text categorization [20], Named entity disambiguation [8], and Entity ranking [9]. Therefore, it is essential that Wikipedia's content is up to date and accurate.

[1] We will be using the terms *references*, *concepts* and *links* interchangeably

Wikipedia volunteers classify articles into seven categories: *FA*, *GA*, *A*, *B*, *C*, *Start* and *Stub*[2] respectively. According to statistics, 88% of the Wikipedia articles belong to the stub and start category. However, given the huge size and growth rate of Wikipedia, a manual revision and maintenance process is unworkable in the long run. The problem is not only limited to Wikipedia but also a critical issue for collaboratively built resources like Open Directory Project (ODP)[3]. Therefore, it is crucial to develop automatic link and content suggestion methods which aid editors discover related content and maintain the rich link structure of Wikipedia.

We enrich Wikipedia articles by recommending hyperlinks. More specifically, we address the following research problem: Given the introductory content, title, and semantics of a Wikipedia article like category and structure, can we suggest concepts which could help editors in discovering content related to them? We recommend links (concepts) and leave the choice of adding content from these links to editors. We envisage that our concept suggestions can be motivation for the new editors to write the articles without necessarily having enough domain expertise in the related topics they are writing or editing.

Our proposed method is motivated by how humans search for information. We search for information related to the stub by formulating queries against a search engine. Our central assumption here is information related to stub article is available on the web and can be retrieved by the search engine. The retrieved web content is used to provide concept suggestions for the stub.

Apart from the other methods which suggest missing links, we also, formalize the three properties of an appropriate reference, namely, suggested references should: (1) be relevant to the stub, (2) cover representative (diverse) subtopics of the stub, and (3) are obtained from authoritative sources on the web. Our proposed method attempts to ensure these properties while providing recommendations. For this work, we have limited our suggestions only to other Wikipedia articles. In summary, our essential contributions are: (1) We propose a novel method for augmenting the hyperlink structure of Wikipedia articles. (2) We propose an automated evaluation method which leverages Wikipedia's revision history.

2 Background

Explicit Semantic Analysis (ESA)

In order to obtain a meaningful interpretation of text, we use Explicit Semantic Analysis (ESA) proposed by (Gabrilovich and Markovitch 2007) [6]. ESA uses Wikipedia as its source of world knowledge and was proposed to estimate semantic relatedness between two text fragments. ESA takes a text fragment as input and returns a list of Wikipedia concepts as output which are weighted by the relevance of the concept to the text. The central hypothesis based on which ESA works is: *each article in the Wikipedia corresponds to a single concept.* An inverted index containing a mapping from words to Wikipedia articles that contain them is pre-built and stored as a preprocessing step.

[2] en.wikipedia.org/wiki/Wikipedia:Stub

[3] en.wikipedia.org/wiki/DMOZ

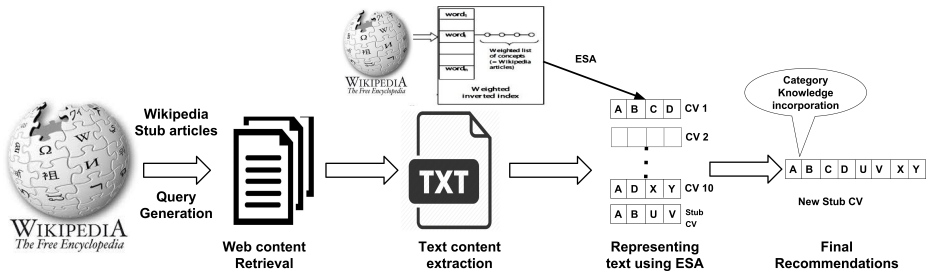


Fig. 1. WikiAug: System architecture.

To obtain the ESA representation of a text fragment, for each word in the text, ESA inverted index is searched and then the corresponding concepts containing that particular word are retrieved. These concepts are combined to form the weighted concept vector, where weights correspond to the TFIDF value of the word in the ESA article corresponding to the concept. The list of concepts are ordered by the weight to get the final ESA representation.

3 Problem Formulation

In this paper, we study the problem of suggesting Wikipedia articles (concepts) that contain relevant information to add to Wikipedia stubs. We formalize the task as follows. Given a Wikipedia stub S and a set of all Wikipedia articles W, we have to decide for each $w_c \in W$ whether it is relevant for improving the content of Stub S. More formally, we have to estimate:

$$P(rel|w_c, S), \tag{1}$$

where rel is the relevance of document w_c with respect to stub S. An article is considered relevant and a valid suggestion if it can enhance the current content of stub S. Note that we are limiting our suggestions to Wikipedia pages but they can be any other web pages providing the description of the stub.

4 WikiAug (WA)

Our proposed approach for link augmentation is called *WikiAug (WA).* Figure 1 shows the system architecture of *WikiAug.* Our approach is motivated by how humans search for information. We observe that it is typical of humans to search for information using the stub title as a *search keyword* or as a *query* against the search engine.

Our central assumption for this work is, information related to the stub is available on the Web and can be retrieved by the search engine. $WikiAug$ system takes a Wikipedia stub title S as input and returns a list of missing Wikipedia concepts L. We achieve this by comparing S against authoritative sources on the Web.

For a given Wikipedia article S, our system deals with two key issues: (1) How do we find authoritative sources for S from the Web? and (2) How do we find missing concepts in S?

As shown in Figure 1, we propose a five-step procedure *WikiAug (WA)* for link augmentation of stubs. These steps are described in the Table 1. Concepts suggested by our method are classified as relevant or irrelevant based on the ground truth links. If a link suggested by our method is present in the ground truth links set it is considered as a *relevant* one else an *irrelevant* one. Our link suggestions for a query are given in the Table 3.

4.1 Definitions

Wikipedia Article Graph (WAG). Is the graph constructed with nodes as articles and edges representing hyperlinks between the Wikipedia articles.

Wikipedia Category Graph (WCG). Wikipedia categories[4] are organized in a taxonomy like structure called the *Wikipedia Category Graph (WCG)* which captures hyponymy or meronymy relations. Wikipedia category may contain subcategories which further contain Wikipedia articles. However, since Wikipedia does not enforce strict guidelines of taxonomy; the graph may contain cycles or disconnected subcategories. However, these cycles or disconnected subcategories do not influence our algorithm's performance since we do not use any graph theory algorithms in our approach.

Wikipedia Subtopics (Aspects) or Structure. We define structure[5] as the representative subtopics information of a stub article (topic). The central idea here is that any Wikipedia article contains similar sections in similar Wikipedia articles. This motivates us to leverage this structure to cover representative subtopics of the stub article (Details in subsection 4.4). For instance, a query like *Donald Knuth* contains similar sections like *Education, Academic life, Research, Awards and achievements* and *Books and Publications* which are similar to sections in other similar articles.

4.2 Content to Capture Context of the Query (Co)

To mitigate the problem of irrelevant recommendations we consider context of the stub article (query). This is the same idea as suggested in ESA (Gabrivolich et al. 2007) [6]. They hypothesize that ESA may perform inherent disambiguation when provided with the sufficient context of the stub query. To obtain context, we represent the stub article text in ESA Concept Vector and obtain top 10 concepts. These concepts are appended to the stub title and this forms the set of reformulated queries given in Table 2 for Web content retrieval (Refer to the Table 2 for examples of reformulated queries).

4.3 Classification Using Category Knowledge (Cat)

Furthermore, analysis of suggestions by WA and WA+Co gave us insights that we need domain knowledge to provide meaningful recommendations. This domain knowledge is available in the form of category labels.

[4] en.wikipedia.org/wiki/Category

[5] en.wikipedia.org/wiki/Help:Section

Table 1. WikiAug method (WA).

1. **Query Generation:** To search for information on the web, we formulate queries using the stub title. For example, for suggesting links to the stub article on *Michael Jordan*. The query formulated using the stub title is *Michael I. Jordan*. Modified queries are formulated to improve efficacy of WA by the below-mentioned approaches. The following modifications are done in this step for WA variants:
   - Title (WA)+Co,
   - Title (WA)+Co+S,
   - Details of. Structure extraction and Category knowledge extraction are explained in Subsection 4.4 and 4.3 respectively.
2. **Web Content Retrieval:** The query generated from the above step is used to fetch top 10 URLs (search results) from a search engine. We use Google for obtaining the results. These results form the set of relevant Web content (external Web articles) used for suggestion,
3. **Text Extraction:** Web content obtained from the previous step is typically available as web pages. Web pages, in general, contain text snippets in between the HTML tags along with the noise. Therefore, it requires cleaning to retain only the relevant information. Removal of irrelevant content is done using Boilerplate detection[10]. This technique classifies the text snippets as relevant if they contain information related to the body of the article and irrelevant if it is website related content (e.g., header and footer or advertisements content present in a webpage),
4. **Representing Text in ESA Concept Vector:** The resultant text from the above step is given as input to ESA. As mentioned earlier, we obtain *Concept vector* or *Meaning vector* using ESA. Each component of this $Concept\ vector$ is the title of a Wikipedia article. We represent both stub text and the text extracted from external web articles in ESA concept vectors,
5. **Final Recommendations:** From the above step, we obtain concept vectors for the stub and the other external Web articles. Now, we compare Wikipedia article concept vector against the web article concept vectors by looking for the concepts covered. The concepts which are absent in the stub but covered in other articles are given as recommendations to the stub.
   - The following step is for variants of WA named as WA+Co+Cat, WA+Cat+S and WA+Co+Cat+S.:
   - In this step, we classify the WA suggestions with the help of Wikipedia category knowledge by posing this as a classification problem. Further details in Subsection 4.3.

We use these category labels as our class labels and our suggestions as the test instances to be classified by the classification task. We approach this classification task by posing it as a *Multi-Class classification* problem. Thus, we hypothesize that: Semantic features like category label available in Wikipedia articles are useful for providing meaningful suggestions. Refer to the Table 2 for examples of reformulated queries. We prefer multi-class classifiers over binary classifiers because they give us a more balanced distribution when compared to merging all non-relevant recommendations into a single category. To perform this classification, we leverage existing work on text classification by (Chang et al. 2008) [5]. This classification is performed without any labelled training data, utilizing the world knowledge available in ESA, and by understanding of the labels (i.e., mapping documents and labels to a semantic space).

Algorithm 1: Structure or aspect extraction.

input : Wikipedia stub category: c, Wikipedia articles: W, parameter: k
output: Wikipedia structure (or aspects): A
$C \leftarrow retrieveArticles(W, c)$
$H_C \leftarrow extractSectionHeadings(C)$
$aggregateSectionHeadings(H_C)$
for *each* $heading(h) \in H_C$ **do**
  $S_C \leftarrow retrieveSections(H_C, h)$
  $A \leftarrow FrequentSections(S_C, k)$
end

This classification is performed in two steps:

1. In the first step, we represent both labels and documents in a semantic space that allows one to compute the semantic similarity between a document (stub) and a potential label.
2. In the second step, we use the similarity computed in the first step to learn a machine learning classifier for classifying recommendations.

4.4 Wikipedia Structure (Aspects) Extraction(S)

As mentioned in section 4.1, Structure (S) contains key pieces of information available in a Wikipedia article. We extract structure with the help of section titles of Wikipedia articles. We accumulate these section titles from similar articles belonging to the Stub category. Our algorithm for structure (aspect) extraction is given in Algorithm 1. We adapted our structure extraction algorithm from the prior work by (Reinanda et al. 2016) [16], (Sauper et al. 2009) [17] and (Banerjee et al. 2015) [2].

As shown in the algorithm, we first, retrieve Wikipedia articles of all the stub queries belonging to their corresponding category c. It can be observed that it is typical of Wikipedia articles that they belong to several categories. All possible categories labels for any Wikipedia page are listed at the bottom of the page. Among the possible categories, we choose the most relevant stub category as input to our Structure extraction algorithm. To select the most relevant stub category, we estimate the ESA semantic similarity between the possible category labels and stub title. The category with the maximum similarity value is chosen as the most relevant one.

Further, we extract the section headings within the articles belonging to the selected most relevant category. For each section heading, we remove the stop words and aggregate the section heading into one section heading, this is our aggregate section headings step. We then take the k most frequent sections as Structure (or Aspects) from the extracted section headings. Refer to Table 2 for reformulated queries.

5 Empirical Evaluation

5.1 Datasets Description

We performed experiments on datasets belonging to the Wikipedia categories *Machine Learning, Physics, Computing, and Environment*.

Table 2. Reformulated queries for the query *Michael I. Jordan*.

Method	Reformulated queries
WA	Michael I. Jordan
WA+Co	Michael I. Jordan and Association for the Advancement of Artificial Intelligence, Michael I. Jordan and Computer scientist, Michael I. Jordan and Daniel Walker Howe, Michael I. Jordan and University of California Berkeley, and Michael I. Jordan and Nonparametric regression
WA+Co+Cat	same as WA+Co
WA+S+Cat	Michael I. Jordan and Education, Michael I. Jordan and Career, Michael I. Jordan and Academic life, Michael I. Jordan and Biography, Michael I. Jordan and Research, Michael I. Jordan and Honors and awards, Michael I. Jordan and Notable Students, Michael I. Jordan and Roles, Michael I. Jordan and Positions of responsibility and Michael I. Jordan and Books and Publications
WA+Co+Cat+S	Combined set of queries from WA+Co and WA+S+Cat

We curated these datasets by extracting articles using the Wikipedia API[6]. Given a Wikipedia category, this API recursively extract articles present in the category hierarchy that can be reached by the crawler using top-down traversal and allows us to download articles in XML format.

We then extract text content and hyperlinks available in these articles using a python script called *WikiExtractor*[7]. Our curated datasets are Wikipedia articles belonging to the years 2008 and 2015 from the above mentioned Wikipedia categories. From each category, we consider only the Wikipedia articles which got promoted to higher quality grades (relatively Non-Stub) in 2015 from Stubs in 2008 as our queries for link augmentation.

5.2 Classification Using Category Knowledge: Experimental Setup

As already mentioned, our classification assumes no training data is available to us, the only information available is the category of the labels. The efficacy of our approach lies in the rich semantic representation like ESA representation. We use the ESA implementation provided in Descartes library.[8]

To evaluate the effectiveness of the ESA representation, we use for each text fragment concept vector representations of size 10, 50 and 100 concepts. We then use K-Nearest neighbour classifier to perform classification and $K = 20$ set empirically for all the datasets mentioned above. All the results presented here are the average of ten trials. Accuracy of the classifier on the test data is *65.7%* using *BOW-NN* and *85.3%* using *ESA-NN*.

Even though we have used ESA for text representation recently there have been approaches for learning Word embeddings like *Word2vec* proposed by (Mikolov et al. 2014) [13] and have been widely accepted in the NLP community. These approaches are for representing words in terms of vectors and the embeddings are typically trained by neural networks. A thorough comparison of the suitability of several different word embeddings for classification without training labels called as *Dataless Classification*

[6] en.wikipedia.org/wiki/Special:Export

[7] medialab.di.unipi.it/wiki/Wikipedia_Extractor

[8] cogcomp.cs.uiuc.edu/software/descartes/descartes-0.2/doc/README.html

Table 3. Recommendations given by WikiAug and its variant methods for the query *Michael I. Jordan*.

* Only a few irrelevant recommendations are shown for the sake of brevity.	
Method	**Recommendations**
WA	**RELEVANT:** Non-parametric regression, Semi parametric regression, Layers, non-parametric, nonlinear, pehong, chen, distinguished, david, heckerman, kearns, marina, meila, Nonlinear system, Non-parametric statistics, Layering, Generalization error
	IRRELEVANT: Susan L. Graham, Chicago Bulls season, Sam Michael, Jordan, James R. Jordan, Sr., Sean Chen (politician), Steve Chen, The Jordan Rules (book), Sean Chen (artist), Chicago Bulls season, Jordan–European Union relations, Rugby union in Jordan **
WA+Co	**RELEVANT:** Association for the Advancement of Artificial Intelligence, Semi-parametric regression, Non-parametric regression, Non-parametric statistics, Regression analysis, Artificial Intelligence: A Modern Approach, Polynomial regression, Statistical model, Non-parametric Bayesian methods, Markov Chain Monte Carlo, non-parametric analysis, probabilistic graphical models
	IRRELEVANT: Daniel Walker Howe, Shauna Howe, Timothy O. Howe, Lord Howe Island Airport, David J. Howe, Howse Pass, Jeremy Howe, Jordan, Jordan-European Union relations, Stephen Woolley, Rational agent, Sam Michael, Jordan, James R. Jordan, Sr., Piero Scaruffi, 1990-91 Chicago Bulls season, Sam Michael, Jordan
WA+Co+Cat	**RELEVANT:** Association for the Advancement of Artificial Intelligence, Semi-parametric regression, Non-parametric statistics, Non-parametric regression, Non-parametric statistics, Artificial Intelligence: A Modern Approach, Polynomial regression, Statistical model, Non-parametric Bayesian methods, Non-parametric analysis, Markov Chain Monte Carlo, Probabilistic graphical models, Non-parametric analysis
	IRRELEVANT: Geoffrey Hinton, Peter Norvig, James R Jordan
WA+Cat+S	**RELEVANT:** Louisiana State University, University of California San Diego, David Rumelhart, Professor, University of California Berkeley, University of California, Berkeley College of Letters and Science, Recurrent neural networks, Bayesian parametric networks, Non-parametric statistics, Non-parametric regression, Non-parametric statistics, Polynomial regression, Statistical model, Non-parametric Bayesian methods, Non-parametric analysis, Markov Chain Monte Carlo, Probabilistic graphical models, Dirichlet Process, ACM-AAAI Allen Newell Award, Association for Computing Machinery, Association for the Advancement of Artificial Intelligence, Artificial Intelligence: A Modern Approach, Andrew Ng, Zoubin Ghahramani, Francis Bach David Blei, Eric Xing, Martin Wainwright, Ben Taskar and Yoshua Bengio
	IRRELEVANT: Kent A Jordan, Peter Norvig, Piero Scaruffi, MIT Center for Collective Intelligence, Jordan, Tom M Mitchell, Miller Institute, Igor Aleksander, Artificial brain, Sigma Xi, Bureau of Intelligence and Research, Edward Feigenbaum, Artificial Intelligence: A Modern Approach, Mike Jackson (systems scientist), Minds and Machines, Sam Michael, John Canny, Geoffrey Hinton, James R Jordan, Sr, Commonsense knowledge base, Seed AI
WA+Co+Cat +S	**RELEVANT:** Louisiana State University, University of California San Diego, David Rumelhart, Professor, University of California Berkeley, University of California, Berkeley College of Letters and Science, Recurrent neural networks, Bayesian parametric networks, Non-parametric statistics, Nonparametric regression, Non-parametric statistics, Artificial Intelligence: A Modern Approach, Polynomial regression, Statistical model, Nonparametric Bayesian methods, Nonparametric analysis, Markov Chain Monte Carlo, Probabilistic graphical models, Dirichlet Process, ACM AAAI Allen Newell Award, Association for Computing Machinery, Association for the Advancement of Artificial Intelligence, Andrew Ng, Zoubin Ghahramani, David Blei, Eric Xing, Martin Wainwright, Yee Whye Teh, Ben Taskar and Yoshua Bengio
	IRRELEVANT: Kent A Jordan, Artificial Intelligence: A Modern Approach, Peter Norvig, Geoffrey Hinton, Piero Scaruffi, Tom M Mitchell, Miller Institute, Igor Aleksander, Artificial brain, Sigma Xi, Bureau of Intelligence and Research, Common sense knowledge base

was given by (Song et al. 2014) [18]. Their comparison study validates that out of all the semantic similarity measures considered for *Dataless classification*, ESA performs better.

Since our ultimate goal was concept recommendations for Wikipedia articles utilising ESA representation seemed straight forward approach as the output of ESA algorithm is set of Wikipedia concepts which can be directly utilized as concept recommendations.

5.3 Link Recommendation Evaluation

We demonstrate the effectiveness of our *WikiAug* method and its variants by evaluating it on Wikipedia articles.

Groundtruth. Wikipedia allows access to complete revision history of its articles. We accessed from the revision history, links which got added to the stub articles over the years 2008 to 2015 and these links form our set of ground truth links. The set of ground truth links is incomplete because many relevant and important links might have been missed but may allows us to gain useful insights.

Evaluation Measures. To evaluate our recommendations, we defined three measures, i.e., Precision (P), Recall (R) and F_1-score. For our evaluation, a true positive recommendation is a link suggested and added to the corresponding 2015 Wikipedia article. In a similar fashion, a false negative and false positive have been defined. Average Precision and Recall for each Wikipedia article are calculated and averaged over the number articles considered for evaluation and the corresponding F_1-score is also calculated accordingly.

Baseline. We propose our own baseline method which takes help of Wikipedia category tree for providing recommendations. As already described in the Wikipedia category graph (WCG), for a category C, there can be subcategories $S_1, S_2 \dots S_n$ and within each subcategory there can be many Wikipedia articles.

Suppose $W_1, W_2 \dots W_n$ belong to subcategory S_1 and W_1 is a Stub article. Our baseline method recommends all Wikipedia articles at the same level (siblings) as W_1 to the stub W_1. That is essentially our baseline method is doing a Depth first traversal on Wikipedia category tree to reach the stub article node and then a Breadth first traversal from the stub node to obtain all the nodes present at the same level as stub. This way we end up obtaining the sibling nodes of stub and these form the set of our baseline suggestions.

The Wikipedia category tree has the property of all similar articles being grouped under the same category or subcategory by the Wikipedia editors (humans). Therefore, articles at the same level will be highly relevant to each other and this leads to strong recall of our baseline method. Although our baseline has a strong recall, it suffers from poor precision, since the diverse information scattered across different subcategories, categories and at different depths is not captured while providing the suggestions.

We compared precision, recall obtained by our baseline method with the methods proposed above. These results are given in Table 4. As observed in Table 4, incorporating category knowledge leads to higher precision. This trend is observed because we do not suggest the links which are irrelevant or farthest in meaning to stub.

Table 4. Precision (P), Recall (R) and F-score (F_1) values obtained from Baseline and WA and it's variants. Highest F-score (F1) values obtained are in Bold.

	Computing			**Machine Learning**			**Physics**			**Environment**		
Method	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Baseline	0.02	0.28	0.03	0.02	0.26	0.04	0.04	0.39	0.07	0.04	0.48	0.07
WA	0.19	0.08	0.11	0.07	0.06	0.06	0.17	0.11	0.13	0.29	0.14	0.19
WA+Co	0.21	0.11	0.15	0.14	0.07	0.10	0.23	0.16	0.19	0.33	0.15	0.20
WA+Co+Cat	0.71	0.16	0.26	0.69	0.15	0.25	0.71	0.17	0.27	0.79	0.21	0.33
WA+Cat+S	0.62	0.46	0.53	0.58	0.34	0.43	0.69	0.59	0.64	0.70	0.62	0.66
WA+Co+Cat+S	0.73	0.48	**0.58**	0.63	0.37	**0.47**	0.72	0.61	**0.67**	0.75	0.64	**0.69**

Table 5. Precision (P), Recall (R) and F-score (F_1) values obtained from (West at al. 2009 - 10)'s approach and WikiAug (WA+Co+Cat+S) approach.

	West et al. (2009)			**WikiAug**		
Category	P	R	F_1	P	R	F_1
Computing	0.15	0.20	0.17	0.73	0.48	0.58
Machine Learning	0.11	0.19	0.14	0.63	0.37	0.47
Physics	0.17	0.26	0.21	0.72	0.61	0.67
Environment	0.23	0.32	0.27	0.75	0.64	0.69

However, this doesn't contribute to increase in recall. To improve recall of our method, we incorporated structure knowledge. This lead to retrieval of diverse links which are effective in covering various aspects of the stub. We further present our quantitative results by comparing our method's precision, recall and F_1-score with the values obtained by (West et al. 2010) in the Table 5.

6 Related Work

The link prediction problem in information networks like Wikipedia is posed in many variants. A comprehensive study of this problem was performed by (Nowell et al. 2007) [11].

Since augmentation of Wikipedia stubs is the broad goal of our work, we have used Wikipedia as our primary dataset for evaluation. In this section, we briefly describe other methods which proposed solutions for link prediction and augmentation in Wikipedia.

6.1 Link Prediction in Wikipedia

In this section, we compare and contrast our work with other approaches for predicting missing hyperlinks from Wikipedia. Automated methods to identify missing hyperlinks from Wikipedia were extensively studied in the past literature [12, 14, 7].

Table 6. Top suggestions given by WikiAug for the query eurozone crisis.

Human added	**West et al. 2009**	**WikiAug**
1. European debt crisis	1. European central bank	1. European debt crisis
2. European Union	2. Inflation	2. Causes of the European debt crisis
3. European Central Bank	3. OECD	3. European Fiscal Compact
4. International Monetary Fund	4. Eurobarometer	4. European Union
5. Fiscal Union	5. Single market	5. Greek government debt crisis
6. European Stability Mechanism	6. Gross domestic product	6. European Stability Mechanism
7. Outright Monetary Transactions	7. European commission	7. Policy reactions to the Eurozone crisis
8. European Financial Stability Facility	8. Motion of no confidence bank	8. Economic and Monetary union of the European union

We classify these methods broadly into two categories based on the type of solutions they provided. The first group of methods proposed by (Mihalcea et al. 2007) [12], (Milne et al. 2008) [14], and (Meij et al. 2012) [7] solved the problem of predicting the missing out-going links from a given Wikipedia article. These methods process arbitrary text documents and predict outgoing links. They leverage the textual content and graph structure of Wikipedia for predicting missing links.

The second group of methods (West et al. 2009) [21], (Adafre et al. 2007) [1], and (Noraset et al. 2014) [15] solved the problem of predicting future outgoing links to Wikipedia articles. These methods take a Wikipedia article as input and utilize already existing links or perhaps text content to predict future outgoing links.

6.2 Content Augmentation

Content Augmentation deals with adding textual content to incomplete or simply stub articles. In this subsection, we briefly describe past literature on content enrichment. (Sauper et al. 2009) [17] proposed solution for the problem of populating Wikipedia pages obtaining content from Web search engines. They extract web content formulating queries with the help of article title and structure extracted from similar articles in a domain.

Recent work by (Banerjee et al. 2015) [2] and (Banerjee et al. 2016) [3] developed systems WikiKreator and WikiWrite which focused on populating content summaries for stub and red-link articles respectively. WikiKreator system proposed by (Banerjee et al. 2015) leverages category knowledge to suggest minimally redundant and more interpretable content summaries. In addition to this, WikiWrite system incorporates coherence.

6.3 Comparison to Previous Methods

To highlight the contributions of this work, we will now compare it with the existing approaches introduced in Section 6. (West et al. 2009) proposed an approach which outperformed the then the state of the art algorithm [14].

Table 7. Left: The 18 novel links (i.e., topics) human editors added to the Wikipedia article about *Computer Programming* between March 2009 and April 2010. Middle: The 18 top suggestions of West algorithm Right: The 18 top suggestions of WikiAug.

Human-added suggestions	West et al., suggestions	WikiAug suggestions
1. Troubleshooting	1. turing completeness	1. Outline of computer prog.
2. U.S. Air force	2. computer science	2. Pascal (prog. lang.)
3. Adacore	3. High - level prog. lang.	3. Callback (computer prog.)
4. Principle of linguistic relativity	4. Java (Prog. lang.)	4. APL (prog. lang.)
5. Card stock	5. Control flow	5. Automatic prog.
6. High - level prog. lang.	6. Haskell (prog. lang.)	6. History of prog. lang.
7. Temporary file	7. UNIX	7.Prog. tool
8. Memory leak	8. Lisp (prog. lang.)	8. Prog. lang.
9. Race condition	9. Ruby (prog. lang.)	9. Java (prog. lang.)
10. Ergonomics	10. Type system	10. Software
11. Maintainability	11. Subroutine	11. Functional prog.
12. Software bug	12. Comparison of prog. lang.	12. Julia (prog. lang.)
13. Vulnerability (Computing)	13. Difference engine	13. Application prog. interface
14. Scripting lang.	14. Python (prog. lang.)	14. Clarion (prog. lang.)
15. Measuring prog. lang. popularity	15. Z3 (computer)	15. Game prog.
16. 1947	16. Halting problem	16. Parameter (computer prog.)
17. The art of computer prog.	17. Abacus	17. Scheme (prog. lang.)
18. Gerhald Weinberg	18. UNIX - Like	18. Extreme prog.

(Adafre et al. 2007)'s method suggest relevant links based on the inlink similarity. Whereas, (West et al. 2009)'s method suggests relevant links based on the outgoing link similarity. Similar to their method we also identify relevant outgoing links. The above two methods capture structural similarity, i.e., the similarity based on the structure of the graph. These methods work with the underlying assumption that *Similar articles contain similar outgoing links*.

They identify similar articles with the number of shared outlinks as a similarity measure. But, this doesn't work effectively for stub articles since do not have the nice properties like strong connectivity to the other Wikipedia articles or sufficient text content. Thus, we devised an orthogonal similarity measure like text similarity (ESA similarity), Domain (Category) information and Structure information.

Previous methods discussed in related work are concept detectors, where as an approach proposed by (West et al. 2010) [22] is a concept (topic) suggester. Similar to theirs, our method is also a concept suggester. Consequently, our method suggestions are not limited to terms or phrases appearing in the current input text but for every possible candidate Wikipedia article. Other than Wikipedia's textual and hyper-textual content, we are exploiting rich semantics utilizing others dimensions like category knowledge and structure available in Wikipedia which previous methods have ignored. We hypothesize:

1. *Category (domain) knowledge is effective to provide suggestion of a concept that is closest in meaning to the stub article,*
2. *Structure of an article is helpful in covering various subtopics to ensure diversity in the article content*.

Efficacy of our approach lies in exploiting these dimensions for providing concept suggestions. While (West et al. 2010) restricted themselves with only outgoing link similarities, we observed that stubs articles contain few outgoing links compared to enriched articles. and it becomes challenging to identify similar articles using only outgoing link similarity. Thus, our proposed dimensions can complement their similarity measure for discovering concepts (or links). We present our qualitative results on a query in Table 6 and 7.

7 Conclusions and Future Work

In this paper, we present a novel approach to enrich Wikipedia articles by suggesting missing links. We provide recommendations using the semantics of Wikipedia articles like category knowledge, content and structure template. We outline implications of the approach beyond link suggestion: It can detect concepts (links) which a Wikipedia article fails to cover.

We believe that this work will inspire the application of similar techniques. As a part of future work, we note that Wikipedia contains different types of links like *External Links*, *References* and *See also*. We intend to investigate how can we modify our approach for providing link suggestions for these type of links.

References

1. Adafre, S. F., Rijke, M.: Discovering missing links in Wikipedia. In: Proceedings of the 3rd international workshop on Link discovery, pp. 90–97 (2005) doi: 10.1145/1134271.1134284
2. Banerjee, S., Mitra, P.: WikiKreator: Improving Wikipedia stubs automatically. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, vol. 1, pp. 867–877 (2015) doi: 10.3115/v1/P15-1084
3. Banerjee, S., Mitra, P.: WikiWrite: Generating Wikipedia articles automatically. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, pp. 2740–2746 (2016)
4. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the web of data. Journal of Web Semantics, vol. 7, no. 3, pp. 154–165 (2009) doi: 10.1016/j.websem.2009.07.002
5. Chang, M., Ratinov, L., Roth, D., Srikumar, V.: Importance of semantic representation: Dataless classification. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI'08, pp. 830–835 (2008)
6. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, pp. 1606–1611 (2007)

7. He, J., de-Rijke, M.: An exploration of learning to link with Wikipedia: Features, methods and training collection. In: 8th International Workshop of the Initiative for the Evaluation of XML Retrieval, pp. 324–330 (2009) doi: 10.1007/978-3-642-14556-8_32
8. Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., Weikum, G.: Robust disambiguation of named entities in text. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP'11, pp. 782–792 (2011)
9. Kaptein, R., Serdyukov, P., de-Vries, A. P., Kamps, J.: Entity ranking using Wikipedia as a pivot. In: Proceedings of the 19th ACM Conference on Information and Knowledge Management, pp. 69–78 (2010) doi: 10.1145/1871437.1871451
10. Kohlschütter, C., Fankhauser, P., Nejdl, W.: Boilerplate detection using shallow text features. In: Proceedings of the Third International Conference on Web Search and Web Data Mining, pp. 441–450 (2010) doi: 10.1145/1718487.1718542
11. Liben-Nowell, D., Kleinberg, J. M.: The link prediction problem for social networks. In: Proceedings of the 12th International Conference on Information and Knowledge Management, pp. 556–559 (2003) doi: 10.1145/956863.956972
12. Mihalcea, R., Csomai, A.: Wikify!: linking documents to encyclopedic knowledge. In: Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, pp. 233–242 (2007) doi: 10.1145/1321440.1321475
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013, vol. 2, pp. 3111–3119 (2013)
14. Milne, D. N., Witten, I. H.: Learning to link with wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM'08, pp. 509–518 (2008) doi: 10.1145/1458082.1458150
15. Noraset, T., Bhagavatula, C., Downey, D.: Adding high-precision links to wikipedia. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 651–656 (2014) doi: 10.3115/v1/D14-1072
16. Reinanda, R., Meij, E., de-Rijke, M.: Document filtering for long-tail entities. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM'16, pp. 771–780 (2016) doi: 10.1145/2983323.2983728
17. Sauper, C., Barzilay, R.: Automatically generating wikipedia articles: A structure-aware approach. In: Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pp. 208–216 (2009)
18. Song, Y., Roth, D.: On dataless hierarchical text classification. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, vol. 28, pp. 1579–1585 (2014) doi: 10.1609/aaai.v28i1.8938
19. Suchanek, F. M., Kasneci, G., Weikum, G.: Yago: A core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, pp. 697–706 (2007) doi: 10.1145/1242572.1242667
20. Wang, P., Domeniconi, C.: Building semantic kernels for text classification using wikipedia. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 713–721 (2008) doi: 10.1145/1401890.1401976
21. West, R., Precup, D., Pineau, J.: Completing wikipedia's hyperlink structure through dimensionality reduction. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 1097–1106 (2009) doi: 10.1145/1645953.1646093
22. West, R., Precup, D., Pineau, J.: Automatically suggesting topics for augmenting text documents. In: Proceedings of the 19th ACM Conference on Information and Knowledge Management, pp. 929–938 (2010) doi: 10.1145/1871437.1871556

Electronic edition
Available online: http://www.rcs.cic.ipn.mx

ISSN: 1870-4069
http://rcs.cic.ipn.mx

Centro de Investigación
en Computación