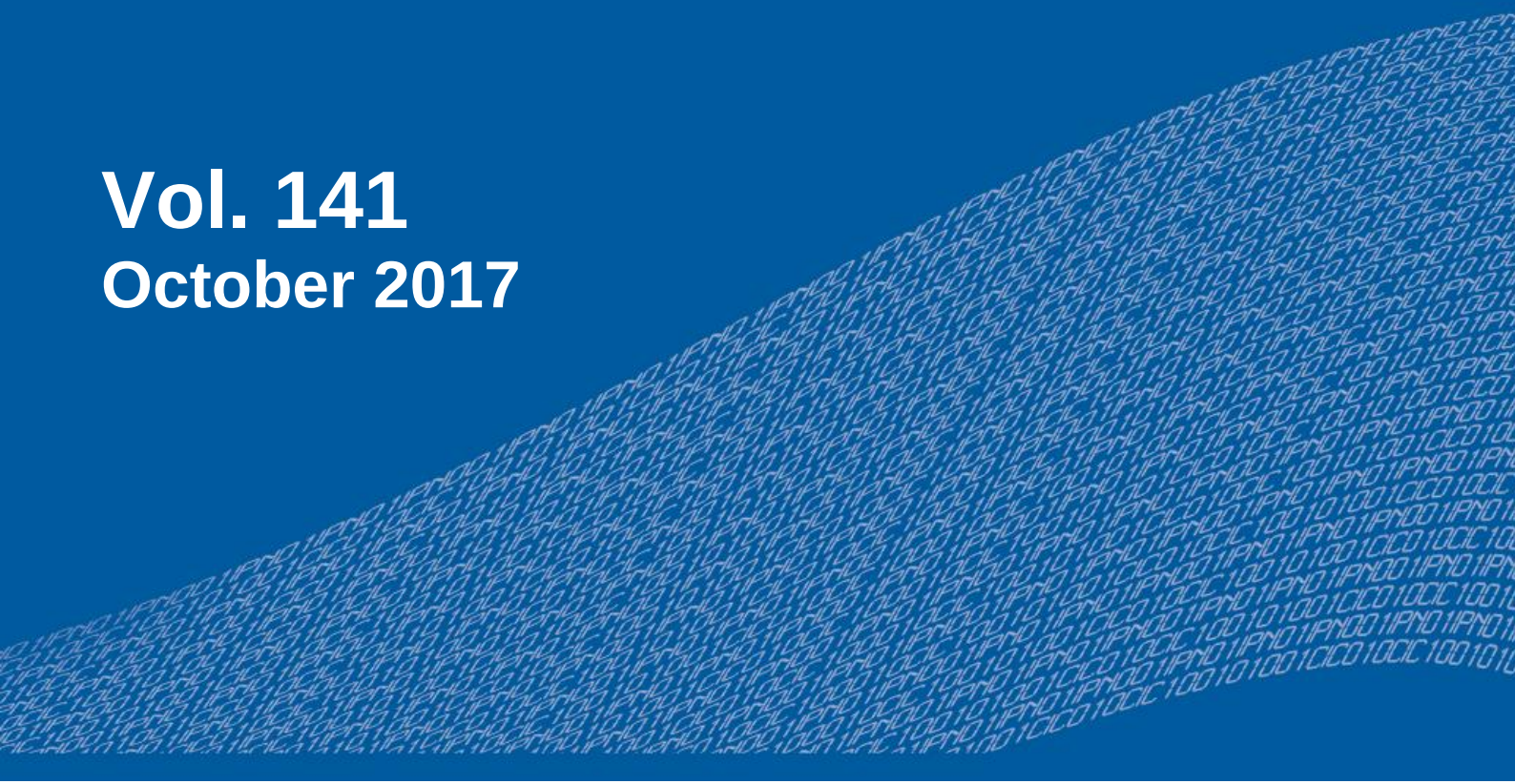
EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

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

Research in Computing Science

ISSN: 1870-4069



Vol. 141
October 2017

Research in Computing Science

Research in Computing Science, Año 16, Volumen 141, octubre de 2017, 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 octubre de 2017.

Research in Computing Science, year 16, Volume 141, October 2017, is published monthly by the Center for Computing Research of IPN.

Volume 141

Advances in Natural Language Processing

Alexander Gelbukh (ed.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2017

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

A Method on Similar Text Finding and Plagiarism Detection based on Topic Model

Li Zhang, Jun Li

University of Science and Technology of China,
School of Information Science and Technology,
China

zlahu@foxmail.com, ljun@ustc.edu.cn

Abstract. This paper proposes a method on similar text finding and plagiarism detection among mass texts, which is based on the LDA mode, when a text need to detect, LDA generates its topics distribution, then calculate the similarity with the text in topics distribution library, we define the most similar text set as SimiSet, plagiarism detection is based on SimiSet, the coming text compares with the text in SimiSet using the method of fingerprint matching. In this paper, we compared several kinds of similarity calculation algorithm; experiment found that the combined algorithm of KNN and JS distance has higher recall rate and lower AIV value on plagiarism detection. We also compared proposed method with the traditional method on plagiarism detection, the result shows our method has higher F1 and smaller search range.

Keywords. Plagiarism, text similarity, LDA, SimiSet, TF-IDF, fingerprint.

1 Introduction

With the rapid development of the computing and Internet technology, the amount of multimedia data including digital text resource increases in a surprising speed, there are many complex relationships among these explosive texts, one of the performances is plagiarism, especially for popular articles and thesis, these misconduct acts like reproducing sections in absence of quoted reference or statements is no longer rare in big data times. To purify the atmosphere of the Internet and academic, it is important to solve plagiarism.

At present, TF-IDF (Term Frequency-Inverse Document Frequency) based on VSM (Vector Space Model) or its modified form is a widely used algorithm in similar text detection, a text's feature weight is expressed by combining term frequency and Inverse document frequency. However, TF-IDF method has several limits, such as ignoring semantic information, sparse problem, especially for short text; sometimes, the word able to characterize text semantics does not have highest weight, a word with a higher weight may has no practical meaning, these problems lead to low computational efficiency.

In order to avoid above problems, we take advantage of LDA [1] (Dirichlet Allocation Latent) topic model to calculate similarity, in which the feature of each text is represented by topic distribution, this representation reveals text semantics and

content information, due to the vector is relatively short, LDA solve high dimensional sparse problem, moreover, calculation procedure has speed up, the computational efficiency has improved. The proposed method of plagiarism detection in this paper is based on one assumption; the texts with plagiarism are similar in some content or topics. The procedure is to find target texts which constitute plagiarism with the source text in similar text set (SimiSet), so the proposed method is two-step method.

2 Related Work

2.1 Similar Text Detection

Similar text detection methods can be divided into three catalogues: VSM based, lexicon based and semantic based, in which lexicon based and semantic based approaches belong to semantic similarity detection [2].

In VSM based methods, the text is divided into many keywords, each weight of the keywords can be calculated by TF-IDF, chi-square statistics (CHI), mutual information and entropy. Guo [3] proposed an improved DF and TF-IDF algorithm, which increase the precision of the similarity calculation by adding keywords to reduce the incorrect filtering of information.

The lexicon-based approach figures the similarity between lexical items by referring to semantic lexicon, WordNet [4] is the most popular semantic lexicon, apart from WordNet, HowNet [5] is a common used Chinese semantic lexicon.

Rada [6] presented a method for measuring text semantic similarity, use corpus-based and knowledge-based similarity measures, experiments show that the proposed method outperforms the method based on simple lexical matching and the traditional vector-based similarity metric.

According to the information gain in corpus, semantic based methods are used to determine the similarity between lexical items. LSI (Semantic Index Latent) [7] and LDA are commonly used. Islam [8] proposed a semantic similarity method based on corpus and improved LCS algorithm, which is suitable for various situations of text representation and similarity discovery, experiment shows that the method has better performance than TF-IDF.TF-IDF cannot utilize semantics, the vector has very high dimensions and is extremely sparse, resulting in inefficient calculation and detection accuracy.

TF-IDF is not suitable for large text set, because of time consuming. The lexicon-based approach utilizes external knowledge, the detection efficiency depends on dictionary quality, which introduces uncertain factor. LSI uses matrix singular value decomposition (SVD) to convert word frequency into singular matrix, although the latent semantic relation is mined, but the parameters increase with corpus size, apart from this, complex calculation and high dimension are another limitation.

Compared with LSI, LDA also uses semantic information, but it can solve high dimension and sparseness problems. There are few parameters in LDA, it is suitable for large corpus.

2.2 Plagiarism Detection

Plagiarism detection methods can be divided into two catalogues: frequency statistic, string matching (also known as fingerprint).

The idea of frequency statistic is to calculate word frequency in two texts, Plagiarism detection become the problem of frequency similarity. Shivakumar developed SCAM [9], the system used VSM and frequency statistic method, and it could solve conflict of intellectual property rights in a certain degree. Si established CHECK [10] system, used the keyword statistic approach to measure text similarity, document structure information is introduced to achieve more accurate detection for the first time. CHENG [11] proposed a method in a mathematic way, the method is based on the representation of Chinese characters in computer, and used frequency statistic to measure text similarity, experiment proved that the algorithm is effective.

The string-matching method selects strings from document, these strings are a sequence of N consecutive words [12], each of which can be represented by a hash value (fingerprint) [13, 14]. Manber implemented Siff [15] tool and presented the approximate concept of fingerprint. Since then, many plagiarism detection systems have adopted this idea, such as COPS [16], KOALA [17], shingling [18], MDR [19], YAP [20]. In addition to the two methods mentioned above, Jamal[21] proposed a method for paragraph plagiarism detection, which is based on semantic analysis and part-of-speech (POS) tagging, to facilitate rapid detection, they put topics, synonyms and POS of paragraphs into the corresponding XML files, experiment showed the method can detect almost all type of plagiarism text.

At present, most of plagiarism detection algorithms are based on frequency statistics and string-matching methods. Statistics method does not consider semantic and text structure, which affects the accuracy of plagiarism detection. Although string matching method also does not use semantic, but it can quickly achieve plagiarism detection, the time and space required of generate fingerprints are relatively small, it can handle more files at the same time, for the case of full plagiarism, the algorithm will detect it quickly. Therefore, we use string matching method for plagiarism detection, experiments showed the method can make quick and accurate judgments for coming detected.

3 Similar Text Detection and Plagiarism Detection Method

3.1 LDA Model and Parameter Estimation

The origin of topic model is from LSI, which considers semantic relations, but it may filter out important features, resulting in poor classification performance. In 2003, Blei proposed LDA, which further improve topic model. LDA has prominent advantages, it has a clear hierarchical structure, is a probability model of word-topic- article layers.

A text has multiple latent topics mixed, every topic is expressed in probability distribution, and therefore text information is transformed into digital information which is easy to model. LDA introduces Dirichlet prior parameters in topic and word layers, both article-topic and the topic-word layer obey polynomial distribution, the parameters do not linearly increase with the growth of training set, which avoids over-fitting, especially appropriate for large corpus.

In this paper, Gibbs sampling is used to estimate model parameters, the ideology is to estimate the topics of each word, then the problem becomes calculating the conditional probability in equation (1), once the topic of each word is determined, the topic-word matrix and the article-topic matrix can be estimated by counting term frequency using equation (2) and (3) [22]:

$$p(z_i = k \mid \overrightarrow{Z_{\neg i}}, \overrightarrow{W}) = \frac{p(\overrightarrow{W}, \overrightarrow{Z})}{p(\overrightarrow{W}, \overrightarrow{Z_{\neg i}})} \propto \frac{n^t_{k,\neg i} + \beta_t}{\sum_{t=1}^{V} n^t_{k,\neg i} + \beta_t} \times \frac{\left(n^k_{m,\neg i} + \alpha_k\right)}{\left[\sum_{k=1}^{K} n^k_m + \alpha_k\right] - 1}, \tag{1}$$

$$\varphi_{k,t} = \frac{n^t_k + \beta_t}{\sum_{t=1}^{V} n^t_k + \beta_t}, \tag{2}$$

$$\theta_{m.k} = \frac{n^k_m + \alpha_k}{\sum_{t=1}^{K} n^k_m + \alpha_k}. \tag{3}$$

z_i denotes topic variables correspond with the i^{th} words, $\neg_i$ denotes the i^{th} excluded term, n^t_k denotes the number of word *t* occurrence in a specific topic *k*, n^k_m denotes the number of topic *k* occurrence in a specific text *m*.

3.2 Text Similarity

The topics generated by LDA model are not always meaningfully, because of lot of noise in training corpus, there are spam topics in topic mining results, especially for large training texts, which affect the performance of text similarity. We remove the stop words, names of individuals, words which only contains a single character, illegal English words, words whose document frequency over 60% and some lowest frequency words.

These words often appear as meaningless words in certain topics, have an impact on subsequent topic distribution extraction. This paper realizes the automatic filtering of spam topic. From the perspective of information theory, the higher the quality of a topic, the word distribution of the topic is more unbalanced, so smaller topic entropy is preferred, use (4) to calculate topic entropy, when the entropy is greater than the preset threshold, then it can be regarded as a spam topic. Suppose the probability distribution of a topic is: $p = \{p_1, p_2, p_3, \ldots, p_n\}$, The entropy of P is:

$$H(p) = H(p_1, p_2, p_{3,} \ldots, p_n) = -\sum_{i=1}^{n} p_i \times \log p_i. \tag{4}$$

We define plagiarism text source collection as CopiedSet, refers to the collection of the text which plagiarism text copied from. for example, assuming that the plagiarism text set only contains text *A*, apart from its original content, the rest is obtained by copying text *B*, *C*, *D*, then *CopiedSet (A)* = {*B, C, D*}. The definition of SimiSet refers to a collection of some most similar text with the coming text, each text is represented by corresponding topic distribution. The common used methods of measuring

differences between two probability distributions are KL distance, JS distance and cosine similarity.

For two probability distributions, $p=\{p_1,p_2,p_3,\ldots,p_n\}$ $q=\{q_1,q_2,q_3,\ldots,q_n\}$:

$$D_{KL}(p,q)=\sum_{j=1}^{n} p_j \frac{p_j}{q_j}. \tag{5}$$

KL distance is not symmetrical, JS distance is symmetric version of KL:

$$D_{JS}(p,q)=\frac{1}{2}\times\left[D_{KL}(p,\frac{p+q}{2})+D_{KL}(q,\frac{p+q}{2})\right]. \tag{6}$$

Cosine similarity is:

$$D_{cosin}(p,q)=\cos\theta=\frac{\sum_{j=1}^{n} p_j\times q_j}{\sqrt{\sum_{j=1}^{n} p_j^2\times\sum_{j=1}^{n} q_j^2}}. \tag{7}$$

Considering that a text may have multiple topics, one text may not only copy another text which is basically the same in the whole topics, but also may plagiarize text which are similar in several topics, such as political and economic texts, A political text may not copy political texts, on the contrary it may plagiarize some economic texts. That is because political and economic texts have certain correlation in some aspects.

If the similarities are measured directly over the entire topic distribution according to (8) and (9), it is possible some text which have high similarity in partial topics are not considered, and therefore the similarity between topic distributions on common non-zero topics must be taken into account.

The modified text similarity measure proposed in this paper fully considers this problem, suppose the topic distribution of coming text *A* is *p*, text *B* is in training corpus, its topic distribution is *q*. We set the topic distribution with very low proportion of probabilities and the spam topic to zero. *sum(A)* represents the sum of the common topic probabilities present in *p*, and *sum(B)* represents the sum of the common topic probabilities present in *q*. The modified text similarity equation is:

$$modifyDist_{JS}(p,q)=\frac{sum(A)\times sum(B)}{D_{JS}(p,q)+const}, \tag{8}$$

$$modifyDist_{cosin}(p,q)=sum(A)\times sum(B)\times D_{cosin}(p,q). \tag{9}$$

const in (9) is a constant to prevent overflow in memory, section 4.4 will compare the effect of four methods indicated by formula (6), (7), (8) and (9).

3.3 Plagiarism Detection Method

In this paper, the string-matching method is used to generate unique values (fingerprints) for a text, we first selected feature by using certain strategies, and then adopted Hash function to generate fingerprints, text *A* is split into several characteristic statements and sentences. $A=\{a_1,a_2,a_3,\ldots,a_m\}$, *Hash(A)* expresses fingerprints of *A*:

$$Hash(A) = \{h_1, h_2, h_3, \ldots, h_m\}. \tag{10}$$

Fingerprint is a mainstream and efficient approach for plagiarism detection, which can not only determine plagiarism, but also can mark details. The main idea of plagiarism detection is matching the fingerprints of two texts, the coincidence times of fingerprints can be expressed by the number of intersection elements of two Hash set, the overlap proportion is ratio of coincidence times to the number of fingerprints, if the overlap proportion is above a certain threshold, it will be determined as plagiarism.

4 Experimental Steps and Results

4.1 Experimental Steps

Our experimental tool is mainly implemented in Java include LDA model, we choose Fudan[1] corpus, which is a Chinese classified corpus, the training corpus contains nearly 10,000 articles in 20 categories.

However, the text distribution is seriously unbalanced, so 11 categories are removed due to containing very few texts, we extract 7480 texts from the remaining corpus, and then filter out short text which contains less than 1000 words, finally the training corpus has 5491 texts. In this paper, firstly, we determined the optimal number of topics for the training corpus and filtered out spam topic, secondly, we given the optimal text similarity calculation method, used the given method to construct SimiSet.

Finally, we compare the proposed method with the traditional method on plagiarism detection. A downsizing dictionary is constructed on the basis of stop words and other meaningless words removing, the size of the dictionary is much smaller than the original one, and so the training set contains less noise and light-weighted, experiment shows that the model contains less spam topics and has better plagiarism detection result.

4.2 Experimental Evaluation

We use three performance evaluation indicators: *F1*, recall rate *R* (Recall), *AIV* (Average Index Value), the indicator *AIV* is defined by us:

$$R = \frac{\left|SimiSet(A) \cap CopyedSet(A)\right|}{\left|CopyedSet(A)\right|}, \tag{11}$$

$$AIV = \frac{\sum Index\left\{SimiSet(A) \cap CopyedSet(A)\right\}}{\left|SimiSet(A) \cap CopyedSet(A)\right|}. \tag{12}$$

Suppose $SimiSet(A) = \{t_1, t_2, t_3, \ldots t_m\}$, $CopyedSet(A) = \{c_1, c_2, c_3, \ldots c_n\}$, *SimiSet (A)* is a set sorted in descending order. *A* may copy more than one piece of text, these texts are added to *SimiSet (A)* in the similarity calculation stage, *R* is the ratio of the number of

[1] http://www.nlpir.org/?action-viewnews-itemid-103

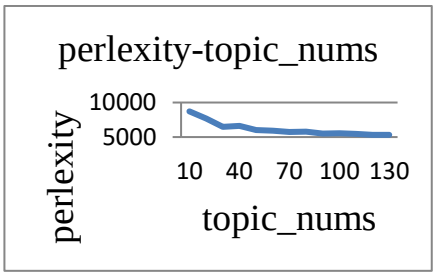


Fig. 1. Perplexity with different topic numbers.

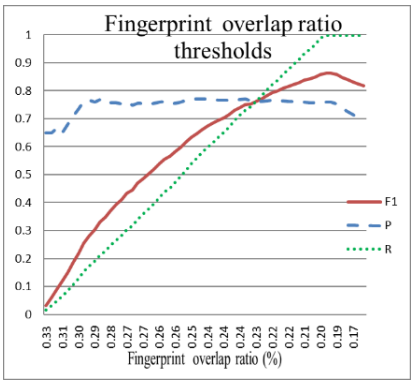


Fig. 2. Fingerprint overlap coincidence.

found plagiarism source text in *SimiSet (A)* and the number of all text in *CopiedSet (A)*, which is formulated in equation (11), operator $|*|$ means obtaining the number of collection elements.

In addition, *AIV* is defined as an average of all index of plagiarism source text in *SimiSet (A)*, which reflects the average rank of plagiarism source text, the higher the rank is, the less fingerprint matching times is used to find plagiarism source text. In equation (12), the operator $Index\{*\}$ indicates the index of plagiarism source text in *SimiSet (A)*.

4.3 Topic Number Selection and Spam Topic Filtering

LDA has few parameters, α, β is super parameter, which have empirical values，K is the number of topics, how to determine the size of K? Blei suggests Perplexity.

To find the optimal K, we draw the curve of perplexity with different number of topics, as shown in Fig.1, perplexity decreases monotonously with the number of topics increased, lower perplexity means better model, but the model is more prone to over-fitting, so we choose the number of topic when the curve begins to converge, for this model K = 100. Experiment indeed achieved good results when K set to 100.

Table 1. *R* compared on four similarity algorithms.

Simset Size	modify_cosin	modify_js	cosin	js
50	0.298	0.257	0.266	0.292
100	0.445	0.430	0.385	0.483
150	0.560	0.563	0.510	0.592
200	0.655	0.655	0.601	0.680
250	0.730	0.732	0.672	0.745
300	0.785	0.806	0.760	0.810
400	0.881	0.890	0.847	0.893
500	0.943	0.933	0.911	0.931
700	0.982	0.973	0.983	0.977
900	0.993	0.991	0.993	0.990
1200	0.996	0.996	0.997	0.994

Table 2. *AIV* compared on four similarity algorithms.

Simset Size	modify_cosin	modify_js	cosin	js
50	19.803	20.587	19.474	20.726
100	37.179	42.020	36.204	41.062
150	54.263	61.456	57.551	56.341
200	71.616	76.839	75.099	71.477
250	87.207	92.368	90.462	84.902
300	100.226	108.699	111.748	100.287
400	127.034	131.311	136.395	123.356
500	148.312	146.073	157.556	136.429
700	166.133	164.285	188.979	157.392
900	172.935	175.356	194.830	164.911
1200	175.192	179.067	197.733	168.108

4.4 Comparison of Several Similarity Methods

The test set consists of 200 plagiarized texts, which copied from 800 source texts, Table 1 shows the trend of *R* when different similarity methods is adopted, these similarity values is calculated by using equation (6), (7), (8) and (9) respectively.

As shown in Table 1, when SimiSet size within [0,700], *R* increases slowly with the SimiSet increases, when the size exceeds 700, *R* converges very close to 1. *R* of the four methods in Table 1 are basically the same except for cosine distance, but *AIV* is different as shown in Table 2. Lower *AIV* value indicates higher rank in SimiSet, it means less number of matches required to achieve the same *R* rate. JS distance is better than other methods, so this paper will use JS distance as text similarity measure.

4.5 Fingerprint Matching Threshold

Different fingerprint coincidence proportion thresholds affect the accuracy of plagiarism detection, to find the appropriate threshold, we also use section 4.4 test set for threshold selection experiment, and the SimiSet size is set to 800. As shown in Fig. 2, with the expansion of the threshold, *F1* value decreases after slowly increase, because the accuracy *P* is less affected by threshold change, the decline of *R* occurs because the higher the threshold, the less the number of plagiarized source texts will be found in SimiSet, so *F1* is mainly affected by *R*. *F1* reaches the peak value 0.863 at the threshold 19.8%.

This shows that the fingerprint method based on string matching has high reliability in finding the plagiarism source text from SimiSet.

4.6 Proposed Method and Traditional Method on Plagiarism Detection

In order to validate effectiveness of the proposed method, we compare it with the traditional plagiarism detection method; in order to find all plagiarism texts, traditional method use one-by-one comparison. Test set in this experiment is from section 4.4. As shown in Table 3, *F1* of the proposed method (0.863) is slightly high then the traditional method (0.828), but our method greatly reduced matching times. The training corpus contains 5,491 texts, the proposed method carries on SimiSet which contains only 800 texts, which is different from traditional plagiarism detection, the number of texts need to match is 14.6% of the original size, so our methods has greatly improved detection speed enhanced accuracy.

If want to further improve *F1* value, one need to make compromise on SimiSet size, which means if SimiSet size is larger, the detection process will be more time consuming, but *F1* value will be promoted. Experiment has found that when SimiSet size is 800, the accuracy and speed can both achieve satisfactory results.

5 Conclusion

This paper introduces the LDA topic model, which solves sparse problem effectively. LDA utilizes semantics of text, which greatly reduces the dimension of text vector representation and enhances its meaning, therefore, the effect of similarity calculation is more accurate and effectively.

In this paper, we find the optimal method to measure text similarity among four similarity algorithms, then we construct SimiSet for similar texts, the size of SimiSet in the experiment is far smaller than the total number of texts in the corpus, the plagiarism source text is mainly concentrated in SimiSet, which effectively reduces the number of texts to match and improves the detection performance and accuracy.

Next, we use fingerprint method to further detect plagiarism details from SimiSet. Finally, we compare our methods with traditional method in plagiarism detection, experiment proof the proposed method is more effective. The research mainly focus on long text, such as thesis, dissertation or web blog, the proposed method is ineffective in short text.

Table 3. Comparison between the proposed method and traditional method.

Methods	collection size	*F1*
proposed method	800	0.863
traditional method	5491	0.828

The subsequent work mainly aimed at short text plagiarism detection and similarity detection, how to effectively distinguish the boundaries between similarity and plagiarism need further study. LDA has many extensions, we will also explore new methods of text modeling and text mining based on LDA.

Acknowledgments. This research was supported by China Academy of Telecommunication Research of MIIT, the project Experimental verification of the basic commonness and key technical standards of the Industrial Internet network architecture. The authors would like to thank the reviewers for their valuable comments.

References

1. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet allocation. Journal of Machine Learning Research, pp. 993–1022 (2003)
2. Gomaa, W. H., Fahmy, A. A.: A survey of text similarity approaches. International Journal of Computer Applications, vol. 68, no. 13 (2013)
3. Guo, Q.: The similarity computing of documents based on VSM. International Conference on Network-Based Information Systems, Springer, pp. 142–148 (2008)
4. Miller, G. A.: WordNet: a lexical database for English. Communications of the ACM, vol. 38, no. 11 pp. 39–41 (1995) doi: 10.1145/219717.219748
5. HowNet: http://www.keenage.com/html/c_index.html
6. Mihalcea, R., Corley, C., Strapparava.: Corpus based and knowledge-based measures of text semantic similarity. American Association for Artificial Intelligence, pp. 775–780 (2006)
7. Dumais, S. T.: Latent semantic analysis. Annual review of information science and technology, vol. 38, pp. 188–230 (2004)
8. Islam, A., Inkpen, D.: Semantic text similarity using corpus-based word similarity and string similarity. ACM Transactions on Knowledge Discovery from Data 2, no. 2, pp. 1–25 (2008) doi: 10.1145/1376815.1376819
9. Shivakumar, N., Garcia-Molina, H.: SCAM: A copy detection mechanism for digital documents. ACM SIGMOD Record, pp. 11–13 (1995)
10. Si, A., Leong, H. V., Lau, R. W. H.: Check: a document plagiarism detection system. In: Proceedings of the 1997 ACM Symposium on Applied Computing, pp. 70–77 (1997)
11. Cheng, Y., Zhang, J.: An algorithm for the illegal copying detection of digital documents. In: International Conference on Natural Language Processing and Knowledge Engineering, IEEE Press, pp. 384–387 (2005) doi: 10.1109/NLPKE.2005.1598767.
12. Lukashenko, R., Šakele, V., Grundspenkis, J.: Computer-based plagiarism detection methods and tools: an overview. In: Proceedings of the 2007 International Conference on Computer Systems and Technologies, ACM, vol. 285 pp. 40 (2007) doi: 10.1145/1330598.1330642
13. Stein, B., Zu-Eissen, S. M.: Near similarity search and plagiarism analysis. From data and information analysis to knowledge engineering, In: Proceedings of the 29th Annual Conference of the Gesellschaft für Klassifikation e.V, pp. 430–437 (2006) doi: 10.1007/3-540-31314-1_52

14. Schleimer, S., Wilkerson, D. S., Aiken, A.: Winnowing: Local algorithm for document fingerprinting. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, pp. 76–85 (2003) doi: 10.1145/872757.87277
15. Manber, U.: Finding similar files in a large file system. In: Winter USENIX Technical Conference, pp. 1–10 (1994)
16. Brin, S., Davis, J., Garcia-Molina, H.: Copy detection mechanisms for digital documents. ACM SIGMOD Record, pp. 398–409 (1995)
17. Heintze, N.: Scalable document fingerprinting. In: Proceedings of the 2nd USENIX Workshop on Electronic Commerce, pp. 3 (1996)
18. Broder, A. Z., Glassman, S. C., Manasse, M. S., Zweig, G.: Syntactic clustering of the Web. In: Proceedings of the 6th International Web Conference, vol. 29, no. 8-13, pp. 1157–1166 (1997) doi: 10.1016/S0169-7552(97)00031-7
19. Monostori, K., Zaslavsky, A., Schmidt, H.: Match detect reveal: finding overlapping and similar digital documents. In: Proceedings of the Information Resources Management Association International Conference, IDEA Group Publisher, pp. 541–552 (2000)
20. Wise, M. J.: YAP3: Improved detection of similarities in computer programs and other texts. In: SIGCSE technical symposium on Computer science education, vol. 28, no. 1, pp. 130–134 (1996) doi: 10.1145/236462.236525
21. Heinrich, G.: Parameter estimation for text analysis. University of Leipzig, Tech. Rep, Hannover (2008)

ISSN 1870-4069

Arab Women in Western Press: Designing News Corpora for Arab Women in British, American, and German News Media during the Arab Spring

Zahra Mustafa-Awad[1], Majdi Sawalha[2],
Monika Kirner-Ludwig[3], Dua'a Tabaza[1]

[1] The University of Jordan,
Department of Linguistics, Faculty of Foreign Languages,
Jordan

[2] The University of Jordan,
Computer Information Department,
School for Information Technology,
Jordan

[3] University at Albany,
State University of New York,
Department of Educational Theory & Practice,
United States

{z.awad,sawalha.majdi}@ju.edu.jo,
mkirner-ludwig@albany.edu, duaa.tabaza@gmail.com

Abstract. This paper reports on designing comparable corpora to investigate the representation of Arab women in Western press during the so-called Arab Spring. Two corpora were constructed by collecting news articles on Arab Women published by most popular news media in the USA, UK and Germany during the period 2010-2015. The first corpus is made up of articles published in English by a British newspaper, *The Guardian* and an American one, *The New York Times*. The second is composed of articles collected from five German news media: *Die Welt*, *Die ZEIT*, *Frankfurter Allgemeine Zeitung*, *Süddeutsche Zeitung*, and *Der Spiegel*. The corpora were managed using Sketch Engine. The significance of our corpora comes from its combining two topics: Arab women and the Arab Spring. Our study emphasizes the political importance of the Arab Spring and highlights the unprecedented roles that Arab women played in the uprisings that characterized it. The results show that there are clear differences between the two corpora in terms of size and emphasis on themes related to Arab women during that period. These results have important implications for researchers and educators in different disciplines including Corpus Linguistics, Discourse Analysis and Mass Communication.

Keywords: Arab women, corpora, news texts, Arab spring.

1 Introduction

News corpora are one type of specialized corpora that can be used for research purposes, natural language processing development, retrieving information, and systems of machine learning. Examples of news corpora include the Reuters' corpus which was launched in 2000, and was composed of 810,000 articles (Rose, Stevenson, & Whitehead, 2002); and Antonio Gulli's corpus which was compiled in 2004 and was made up of more than 1,000,000 news articles (Gulli, & Signorini 2005).

Also, there are the BBC two datasets which were created to cover news from 2004 to 2005, one consists of 2225 news stories published on various topics and the other is made up of 737 articles taken from their sport website. In addition, there is The New York Times Annotated Corpus (Sandhaus, 2008) which was launched in 2008 and was composed of 1.8 million news articles published between 1987 and 2007.

News corpora have become the basis for many studies on the portrayal of various social groups in media. For example, Baker, Gabrielatos, & McEnery (2013) used a four-million-word corpus to explore the linguistic patterns of representing Muslims in British newspapers. Their results indicate that the portrayal of Muslims and Islam in British press was generally negative.

Also, Al-Hejin (2012) analyzed two huge corpora to compare between the image of Muslim women, including Arab women, in news articles published during the period 2001 and 2007 by BBC and Arab News. He found that they are represented differently in the two corpora in terms of semantic macro-structure; while CNN concentrates on their depiction in contexts of crime and conflict, Arab News provides various shades of their portrayal in a wide range of situations combining their achievements and concerns.

A lot of attention has been given lately to the image of Arab women in media. This special interest is due to their active involvement in the political unrest in the Middle East referred to as the Arab Spring, which has led to dramatic consequences and even to military conflicts in the region FIDH, (2012); and to their featuring in the worldwide media coverage of those events (Pew Research Center, 2012).

However, most research on the issue has focused on content analysis, which highlighted aspects of their revolt, modernity and empowerment (Al-Ali 2012; Dastgeer and Gade 2016; Kelek 2012; Kratochwil 2012; Sjoberg and Whooley 2015). In studies concentrating on linguistic inquiry, Arab women have been dealt with primarily using corpora on Muslim women (Özcan, 2013; Al-Hejin, 2014, Samie and Sehlikoglu, 2015). There is, so far, no evidence of using comparable corpora to study the depiction of Arab women in the press during the Arab Spring, using semantic analysis.

This study describes compiling and analyzing two comparable corpora on Arab women in English and German news during the Arab Spring (2010 - 2015). Each corpus is specialized consisting of news articles collected from the most popular print and online news media in the USA, the UK, on one hand, and Germany, on the other. The purpose of creating the corpora is to analyze the representation of Arab women in Western press during that period using Corpus Linguistics (LC) and Critical Discourse Analysis (CDA) approaches.

The importance of research based on such news corpora comes from the importance of news per se: the perceptions and the attitudes people form through their exposure to media which could lead to taking actions with unfavorable consequences.

Table 1. English articles used in the EAWC according to newspaper.

	Number of articles	Percentage	Number of words	Percentage
The Guardian	505	61.66%	435,410	58.92%
New York Times	314	38.34%	303,554	41.08%
Total	**819**	**100%**	**738,964**	**100%**

2 The Design of the Arab Women Corpora

In preparation of the English and German datasets to build the corpora, any standard links non-specific or irrelevant to the individual articles were excluded from the test versions. Also, all articles were sequentially numbered and double checked for their relevance to Arab women only. All texts were uploaded to Sketch Engine on 16/2/2017.

2.1 The English Arab Women Corpus (EAWC)

The English Arab Women Corpus (EAWC) was created by searching the websites of *The Guardian* and *The New York Times* for relevant articles during the period of October 4, 2010 to June 30, 2015. This period was considered in the study although, historically, the uprisings started in December 2010 and began to fade out in mid-2012. However, protests have continued to take place in a number of Arab countries for several subsequent years and the struggle for power is still going on in some of them. Search terms include in addition to "*Arab women/woman*", the nationality pertaining to each Arab country followed by "*women/woman"*, e.g., "*Egyptian women", "Iraqi women",* etc.

We included all the Arab countries in the search terms because the so-called Arab Spring had its effects spread across the Arab world, although the major events took place in Tunisia, Egypt, Yemen, Libya and Syria. The terms were searched in the headlines and the bylines of the articles for two reasons; the first is to make sure that Arab women were the main topic of the news story, and the second is to identify the nationality of women referred to by personal names in the headline through looking at the byline, which provides it in most cases.

For example, Tawakkul, a proper noun, was one of the frequent key words that appeared with a high significance score. It refers to a Yemeni activist who was one of the three winners of Nobel Prize for Peace in 2011; her nationality was, in many instances, mentioned in the byline of the news articles on the topic.

The total corpus consisted of 819 articles made up of 738,964 words. Table [1] shows the distribution of articles and number of words used to build up the EAWC. It is obvious from the figures in Table 1 that the British newspaper, *The Guardian*, had a higher coverage of Arab women during the Arab Spring than the American newspaper, *The New York Times*, in terms of both number of articles (505 / 314), and word counts (435,410 / 303,554), respectively.

Table 2. German articles used in the GAWC according to news medium.

Newspaper	Number of articles	Percentage	Number of words	Percentage
Die Welt	51	15.84%	37,380	17.02%
Süddeutsche Zeitung (SZ)	76	23.60%	41,895	19.07%
Die ZEIT	59	18.32%	57,466	26.16%
Frankfurter Allgemeine Zeitung (FAZ)	78	24.22%	39,738	18.09%
Spiegel	58	18.01%	43,155	19.65%
Total	**322**	**100%**	**219,634**	**100%**

2.2 The German Arab Women Corpus (GAWC)

The German Arab Women Corpus (GAWC) was compiled by searching the websites of *Die Welt*, *Süddeutsche Zeitung* (SZ), *Die ZEIT*, *Frankfurter Allgemeine Zeitung* (FAZ) and *Der Spiegel*. To collect related articles, expressions searched for were *arabische Frauen (Arab women), tunesische Frauen (Tunisian women),* etc. Again, all Arab countries were included in the search and the concentration was on articles which mention Arab women of various nationalities in their headlines or bylines for the same reasons given above.

The GAWC is made up of 322 articles consisting of 219,634 words. Table [2] shows a summary of the number of articles and word counts in the GAWC. It is clear from Table 2 that there are variations in the number of articles and the word counts published by German newspapers on Arab women during that period. Although SZ and *FAZ* published more articles on the topic (76, 78), respectively; than *Die Welt*, *Die ZEIT* and *Der Spiegel* (51, 59, 58), respectively; *Die ZEIT*'s coverage was the most extensive in terms of the number of words (57,466).

A closer look at Tables 1 and 2 shows that the size of the two corpora is not very big in relation to other conventional linguistic corpora, as we included only such news stories that deal with Arab women as their main topic.

The two tables also indicate that there is a big difference between the two corpora with regard to size; the English corpus is three times larger than the German one. This could be due, among other reasons, to the tendency of German media to generally include Arab women in their coverage of Muslim women (cf. Mustafa-Awad & Kirner-Ludwig, 2017).

2.3 The format and Annotation of the Corpora

Our two corpora were grammatically tagged using Sketch Engine, which adopts the Penn Treebank tagset with slight modifications, with total of 58 tags. For the purpose of this paper we concentrated on the main seven parts of speech, i.e. verbs, nouns, adjectives, adverbs, conjunctions, pronouns, and prepositions.

The corpus was formatted to XML and uploaded to Sketch Engine. Provided metadata include the title of the article, newspaper name, author, date of publication, and section on the website. A sample appears below:

"Perceptions of Arab women have been revolutionised" the Guardian,11 March, 2011

<title>Perceptions of Arab women have been revolutionised</title>
<newspaper>the Guardian</newspaper>
<author>Soumaya Ghannoushi,</author>
<section>Arab and Middle East unrest</section>
<Arab Country> All countries </Arab Country>

<story.date>Friday 11 March 2011 20.30 GMT</story.date>

Fig. 1. A sample of XML format of the arab women corpora.

Table 3. Top 15 most frequent words of the arab women corpora.

	a. English Arab Woman Corpus		**b. German Arab Woman Corpus**	
No	**Lemma**	**Frequency**	**Lemma**	**Frequency**
1	woman	8673	Frau ‘woman’	2350
2	right	1681	Mann ‘man’	655
3	Arab	1652	gegen ‘against’	552
4	man	1628	arabisch ‘Arab’	450
5	Saudi	1604	Mädchen ‘girl’	403
6	work	1058	jung ‘young’	303
7	girl	1028	Saudi-Arabien ‘Saudi-Arabia’	300
8	against	1026	dürfen ‘be allowed to’	281
9	Egypt	980	Ägypten ‘Egypt’	258
10	force	801	Kopftuch ‘headscarf’	255
11	law	800	Kind ‘child’	250
12	political	689	stehen ‘to stand’	246
13	rape	688	Mensch ‘human being’	229
14	activist	662	tragen ‘wear’	223
15	drive	654	islamisch ‘Islamic’	216

3 Data Analysis

The Arab Women Corpora (AWC) were analyzed using Sketch Engine (Killgariff 2014) to generate frequency word lists, concordance, collocations, word sketches and sketch differences. For the purposes of this study, stop words were not considered in the analysis. Samples of (AWC) analyses are given below.

3.1 Frequency Lists of the Arab Women Corpora

The English Arab Women corpus consists of 738,964 words sorted in 28,198 different lemmas; the most frequent (15) are shown in Table 3 a. On the other hand, The German Arab Women Corpus is made up of 221,898 words distributed into 26,090 different lemmas; the top 15 are given in Table 3 b.

Table 4. Top 15 collocations of the Lemmas *Woman* and *Frau* from arab woman corpora.

a. English Arab Woman Corpus				b. German Arab Woman Corpus		
No	**Collocates with 'woman'**	**Freq.**	**MI Score**	**Collocates with *Frau(en)***	**Freq.**	**MI Score**
1	Arab	352	10.15	Jung '*young*'	113	10.45
2	right	356	10.15	Muslimisch '*Muslim*'	44	9.16
3	young	160	9.13	Saudisch '*Saudi*'	36	8.89
4	Allow	89	8.31	gegen '*against*'	31	8.45
5	Group	39	7.05	Recht '*right*'	22	8.16
6	Face	35	6.98	arabisch '*Arabic*'	21	8.09
7	driver	23	6.40	verschleiert '*veiled*'	19	8.00
8	appoint	20	6.22	Ägyptisch '*Egyptian*'	17	7.87
9	activist	14	5.62	fahren '*drive*'	15	7.66
10	veil	12	5.48	Rolle '*role*'	14	7.54
11	participate	12	5.47	Aufstand '*rebellion*'	12	7.35
12	prison	12	5.45	spielen '*play*'	10	7.09
13	free	8	4.88	marokkanisch '*Moroccan*'	9	6.59
14	Egypt	8	4.75	Unterdrückung '*oppression*'	8	6.59
15	honourable	5	4.23	arbeiten '*to work*'	7	6.53

A comparison between Table 3 a and b shows that 10 of the top 15 lemmas in the two sets are equivalent, e.g., *woman/Frau, man/Mann, Arab/arabisch, country/Land, against/gegen*. As for the rest of the lemmas in each set, they appear only in one corpus but not in the other.

For instance, right (1682), work (1081), force (801), law (800) rape (688), and activist (662) feature only in EAWC. On the other hand, *sagen* 'to say, speak' (850), *neu* 'new' (289), *dürfen* 'be allowed to' (281), *Kopftuch* 'headscarf' (255) and *stehen* 'to stand' (246) occur only GAWC.

These results indicate that there were discrepancies in emphasis on issues related to Arab women during that period by the news outlets under investigation. While the English language news media focus on issues like rights, violence, oppression and activism, the German ones concentrate on revolution, restriction, dress code and religion.

3.2 Collocations

When searching for collocations of the lemma *woman* in EAWC, the attribute was set as lemma in the range from 5 to 5. The top 15 collocations are presented according to their frequencies and MI (Mutual Information) scores in Table 4a. In a parallel approach, we conducted the same search for collocations of the lemma *Frau* (i.e. including *Frauen* pl.) in GAWC. The top 15 lexical collocations are shown in Table 4b.

We can clearly see from Table 4 below that although a number of the top collocates of *woman* in EAWC have equivalents in GAWC, others do not, despite the overlap between the two corpora in certain instances. Common collocates stress issues like rights and dress code. For instance, women's rights are exhibited by the collocation of

Table 1. Ethnic and national distinctions.

a. English Arab Women Corpus				**b. German Arab Women Corpus**		
#	**Lemma**	**Freq**	**Score**	**Lemma**	**Freq**	**Score**
1	Somalia	277	81.9	*saudi-arabisch*	81	53.6
2	Sudanese	282	69.4	*Qatar*	26	49.7
3	Lebanese	154	63.9	*Ägypterin*	35	41.1
4	Saudis	102	62.5	*Bahrein*	9	35.8
5	Yemen	370	60.7	*Mauretanien*	15	34.8
6	Tunisian	239	60.3	*Tunesierin*	18	32.7
7	Mauritania	48	50.5	*Oman*	14	28.6
8	Palestinian	212	46.7	*marokkanisch*	36	28.4
9	Libya	290	41.6	*Iraks*	7	28
10	Egyptian	651	40.2	*Somaly*	6	24.2
11	Moroccan	120	34.6	*Jemen*	81	22.9
12	Syrian	468	33.6	*Dubai*	34	21
13	Iraqi	270	33.2	*Palästinenserin*	8	15.6
14	Jordanian	148	32.7	*südsudanesisch*	6	15.3
15	Djibouti	25	29.7	*Jordanien*	26	13
16	Qatar	207	21.9	*libanesisch*	25	12.5
17	Algeria	67	18.9	*libysch*	31	11.9
18	Oman	68	14.8	*Sudan*	22	11
19	Kuwait	69	10.9	*Algerien*	19	10
20	Bahraini	49	9.9	*Kuwait*	9	9.5
21	Emirates	75	8.9	*Syrerin*	5	8.9

right (356, IM 10.15) with *woman/women* and *Recht* (22, 8.16) with *Frau(en).* One of these rights that occurred in the two corpora is driving expressed by *driver* (23, 6.40) in EAWC and with *fahren* 'drive' (15, 7.66) in GAWC, respectively. As for dress code, references are made to *veil* (12, 5.48) and *verschleiert* 'veiled' (19, 8.00) in the latter. As for collocates of *women* that occur only in the EAWC, they focus on empowerment, activism, struggle and restriction. For example, empowerment is expressed by *appoint* (20, 6.22) and *free* (8, 4.75), and activism by *participate* (12, 5.4) and *activist* (14, 5.62).

On the other hand, collocates of *Frau* in GAWC concentrate on religion, rebellion, nationality and oppression. The frequent collocation of *muslimisch + Frau(en)* suggests that the attribute of being *Muslim* is inherently and prototypically tied in with the image of Arab women.

There are also a number of German collocates that explicitly and implicitly point to the uprisings, e.g., *Aufstand* 'rebellion, uprising' (12, 7.35). As for *oppression*, it is expressed by *Unterdrückung* 'oppression' (8, 6.59). Another theme that emerges through the collocation with Arab women in GAWC in nationality. This includes countries where the uprisings took place, such as *ägyptisch* '*Egyptian*' (17, 7.87), as well others where they did not, e.g., in Morocco (*marokkanisch* 'Moroccan' 9, 6.95) and Saudi Arabia (*saudisch* 'Saudi' 36, 8.89).

Table 2. Keywords of arabic women corpora.

a. English keywords			**b. German keywords**		
lemma	**Freq**	**Score**	**lemma**	**Freq**	**Score**
Sunni	82	53.5	*Yeziden*	17	66.7
uprising	198	43.6	*Jungfräulichkeitstest 'virginity test'*	13	45.9
apostasy	54	37.8	*Muslimin 'Muslim woman'*	91	31.9
refugee	390	30.1	*Salafistische 'Salafist'*	7	28
sectarian	62	28.2	*Verhüllung 'veil'*	16	27.7
Shiite	40	28.1	*Hidschab 'hijab'*	16	27.7
demonstrator	69	25.7	*Frauensport 'female sports'*	16	26.3
displace	102	22.1	*Friedensnobelpreis 'peace nobel prize'*	44	24.8
fundamentalist	35	21.8	*Verheiratung 'forced/planned marriage'*	8	23.7
Islamic	556	20.4	*Religionspolizei 'religious police'*	19	22.5
virginity	101	20.2	*Geschlechtertrennung 'gender segregation'*	24	21
famine	25	19.1	*Fahrverbot 'driving ban'*	38	20.7
Yazidi	52	18.8	*Aufstand 'rebellion, uprising'*	60	19.3
fighter	244	18.6	*Demonstrantin 'female demonstrator'*	25	18.5
Coptic	29	18.1	*Kinderehe 'child marriage'*	6	18
Salafist	27	18	*Nikab 'niqab'*	12	17.3
revolution	491	17.9	*Unverschleiert 'unveiled'*	14	17.2
pro-democracy	18	17.7	*Bloggerin 'female blogger'*	23	17
protester	274	16.7	*Nichtmuslim 'non-Muslim'*	6	14.3
secularist	21	15.1	*Sklaverei 'slavery'*	25	13.9
hymen	11	11.1	*Genitalverstümmelung 'genital mutilation'*	29	13.8
Awakening	10	10.2	*Kopftuch 'headscarf'*	255	13.7
extremist	91	10.1	*Frühling 'spring'*	83	13.2
polygamy	27	9.9	*Frauenrecht 'female right'*	71	13.2
demonstration	177	9.9	*Pilotin 'female pilot'*	16	11.9
enslave	20	9.8	*Frauenrechtsgruppe 'female rights group'*	8	11.7
condemnation	31	9.7	*Christin 'female Christian'*	27	11.4
non-Muslim	13	9.5	*Unislamisch 'non-islamic'*	7	10.7
circumcision	23	9.5	*Aktivistin 'female activist'*	86	10.4
sit-in	22	9.4	*Inhaftierung 'arrest'*	9	9.8

a. English keywords			**b. German keywords**		
lemma	**Freq**	**Score**	**lemma**	**Freq**	**Score**
devout	19	9.3	*Angreiferin 'female attacker'*	6	9.7
niqab	39	8.9	*Polygamie 'polygamy'*	9	9.5
harasser	14	8.5	*Wahlrecht 'right to vote'*	29	9.4
Liberation	14	8.5	*Sunnit*	8	9
activist	662	8.5	*Erlaubnis 'permission'*	25	8.9
headscarf	37	8.3	*Massenvergewaltigung 'mass rape'*	9	8.7
blogger	64	8.3	*Modernisierung 'modernization'*	15	7.8
dissent	39	8.2	*Unterdrückung 'oppression'*	29	7.5
migrant	66	8.2	*Patriarchalisch 'patriarchic'*	12	7.4
hijab	88	8.1	*Drangsalieren 'to bully, to harrass'*	5	7.1

The results suggest that, although the news media under investigation reported on women's revolt, activism and protest in the countries where the uprisings were taking place, they also addressed the concerns of women in all Arab countries such as oppression, rights and empowerment.

The results also show that the press coverage varies in its emphasis on the various themes related to Arab women depending on many factors including place and language of publication.

3.3 Key Words

In order to get a fuller picture of the themes related to Arab women reflected in the news during the Arab Spring, we decided to investigate the key words in the two datasets. To that end, we uploaded two reference corpora, one for our English corpus and the other for the German one.

The generated key words list show high frequency and significance scores of terms referring to various Arab nationalities including ones beyond those in which the uprisings took place. This is why we display the key words indicating nationality in a separate table, i.e. Table 5.

As for other significant key words including nouns, adjectives and verbs, they are given in Table 6 below. As both datasets show, religious sectarianism is one salient topic coming up. For instance, in the English corpus we find Sunnis and Shiite with frequencies of 82 and 40 and scores of 53.5 and 28.1, respectively.

This is in addition to Islamic, which occurs in baffling high numbers, i.e. 556 times, with a score of 20.4. Similarly, in the German data, *Yeziden* scored highest with 66.7 (despite its relative low frequency at 17). Within similar ranks, we have *Muslimin* 'Muslim woman' with a frequency of 91 and a score of 31.9.

Although both datasets point to dress code as a common issue, the occurrence of words indicating or referring to this matter are more significant in the German corpus than in the English one. For instance, both *Verhüllung* and *Hidschab* occur at the same score of 27.7 in the German data, while headscarf and hijab occur only at 8.3 and 8.3,

respectively, in the English data. All in all, the German data seems to point more to issues of women's rights or their oppression.

This manifest itself in e.g., lemmas such as *Jungfräulichkeitstest* 'virginity test' with a score of 45.9, *forced marriage* (23.7), *driving ban* (20.7) and *Religionspolizei* 'religious police' with (22.5). *Genitalverstümmelung* 'genital mutilation' (13.8) also falls into this group of lemmas. This aligns with our findings on collocates.

In the English corpus, these issues do not score as high. For instance, virginity scores at 22.2, hymen at 11.1, polygamy at 9.9, slave at 9.8, and circumcision at 9.5. When it comes to women's activism, this is highlighted more in the English data than in the German one, where e.g., *Aufstand* 'rebellion, uprising' (60, 19.3) and *Demonstrantin* 'female demonstrator' (25, 18.5). On the other hand, their English equivalents "uprising" occur at (198, 43.6) and demonstrator at (69, 25.7).

4 Evaluation

Our corpora are clearly different from other comparative corpora used in previous studies in the sense that they combine two topics: Arab women and The Arab Spring. They can be of a great value for researchers and educators in various disciplines including Corpus Linguistics, Discourse Analysis and Mass Communication.

As for research, they can be used for conducting comparative studies between the images of Arab women of different nationalities in Western press, as it includes news articles about women of all Arab countries.

The sub-corpora within the EAWC can be utilized for studying the differences between the newspapers published in English. Similarly, sub-corpora constituting the GAWC can be used for studies on the different German newspapers. Our corpora can be also compared to corpora of news reporting on Arab women during the Arab Spring in other languages including Arabic.

In addition, Arab Spring researchers can find a rich amount of news for content analysis when it comes to women. Journalistic styles between English and German can be compared as well. As for educators, our corpora can be a rich resource for professors and students of various disciplines to draw from for illustration and analysis.

5 Conclusion

It can be concluded that there are clear differences in the news coverage of Arab women during the Arab Spring with regard to the language of publication, the intensity of reporting, the focus on themes associated with them, and stressing their nationalities. The results show that American and British press have covered the Arab Spring more extensively than the German news media, which is reflected in the big difference in the size of the two corpora.

The findings also suggest that there is variation in highlighting general issues related to Arab women such as rights and specific ones associated with certain nationalities like driving. These results have important implications for researchers and educators in different disciplines including Corpus Linguistics, Discourse and Analysis and Mass Communication.

References

1. Al-Ali, N.: Gendering the arab spring. Middle East Journal of Culture and Communication, vol. 5, no. 1, pp. 26–31 (2012) doi: 10.1163/187398612X624346
2. Al-Hejin, B.: Covering muslim women: A corpus-based critical discourse analysis of the BBC and Arab News (Doctoral dissertation, Lancaster University) (2012)
3. Baker, P., Gabrielatos, C., McEnery, T.: Sketching muslims: A corpus driven analysis of representations around the word 'muslim' in the british press 1998–2009. Applied Linguistics, vol. 34, no. 3, pp. 255–278 (2013) doi: doi.org/10.1093/applin/ams048
4. Bardici, M. V.: A discourse analysis of the media representation of social media for social change-The case of Egyptian revolution and political change. Malmö Haögskola/Kultur Och Samhälle, Digitala Vetenskapliga Arkivet (2012)
5. Greene, D., Cunningham, P.: Practical solutions to the problem of diagonal dominance in kernel document clustering. BBC News Datasets, vol. 148. pp. 377–384 (2006) doi: 10.1145/1143844.1143892
6. Gulli, A., Signorini, A.: The indexable web is more than 11.5 billion pages. In: Special Interest Tracks and Posters of the 14th International Conference on World Wide Web, ACM, pp. 902–903 (2005)
7. Dastgeer, S., Gade, P. J.: Visual framing of muslim women in the arab spring: Prominent, active, and visible. International Communication Gazette, vol. 78, no. 5, pp. 432–450 (2016) doi: 10.1177/17480485166402
8. Federación Internacional por los Derechos Humanos: Women and the Arab spring: taking their place? (2016)
9. Kelek, N.: Die arabische revolte und die frauen - eine reise durch ägypten, tunesien und marokko. köln: kiepenheuer & witsch (2012)
10. Kratochwil, G.: Die neuen arabischen Frauen: Erfolgsgeschichten aus einer Welt im Aufbruch. Zurich: Orell Füssli Verlag (2012)
11. Jaworska, S., Krishnamurthy, R.: 2012-On the F word: A corpus-based analysis of the media representation of feminism in British and German press discourse, 1990–2009. Discourse & Society, vol. 23, no. 4, pp. 401–431 (2012) doi: 10.1177/0957926512441113
12. Kandil, M. A.: The Israeli-Palestinian conflict in american, arab, and british media: corpus-based critical discourse analysis (2009)
13. Kilgarriff, A.: The Sketch Engine: ten years on. Lexicography, vol. 1, no. 1, pp. 1–30 (2104)
14. Mohanty, C. T.: Under western eyes: Feminist scholarship and colonial discourses. Feminist review, vol. 30, no. 1, pp. 61–88 (1988) doi: 10.1057/fr.1988.42
15. Özcan, E.: Lingerie, bikinis and the headscarf: visual depictions of muslim female migrants in German news media. Feminist Media Studies, vol. 13, no. 3, pp. 427–442 (2013) doi: 10.1080/ 14680777.2012.712382
16. Farooq, S., Sehlikoglu, S.: Strange, incompetent and out-of-place: Media, muslim sportswomen and London 2012 olympics. Feminist Media Studies, vol. 15, no. 3, pp. 363–381 (2015)
17. Rose, T., Stevenson, M., Whitehead, M.: The reuters corpus volume 1-from yesterday's news to tomorrow's language resources. In: Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02), vol. 2, pp. 827–832 (2002)
18. Santaemilia, J., Maruenda, S.: The linguistic representation of gender violence in (written) media discourse: The term 'woman' in Spanish contemporary newspapers. Journal of Language Aggression and Conflict, vol. 2, no. 2, pp. 249–273 (2014)
19. Signal Media: The signal media one-million news articles dataset (2015)
20. Sjoberg, L., Whooley, J.: The Arab spring for women? representations of women in middle east politics in 2011. Journal of Women, Politics & Policy, vol. 36, no. 3, pp. 261–284 (2015) doi: 10.1080/1554477X.2015.1050902

ISSN 1870-4069

Mapping Dependency Relations onto Semantic Categories

Cătălina Mărănduc[1], Monica Mihaela Rizea[2], Dan Cristea[3]

[1] Al. I. Cuza University, Faculty of Computer Science,
Academic Linguistics Institute I. Iordan – Al. Rosetti Bucharest,
Romania

[2] University of Bucharest,
Solomon Marcus Center for Computational Linguistics,
Romania

[3] Al. I. Cuza University, Faculty of Computer Science,
Institute for Computer Science, Romanian Academy - Iasi Branch,
Romania

{catalina.maranduc,dan.cristea}@info.uaic.ro,
monicamihaelarizea@gmail.com

Abstract. The paper focuses on a dependency treebank that aims to illustrate the Romanian language in more styles of communication and in more geographical and historical variants. The treebank, called UAIC-RoDiaTb, contains 18,630 sentences and it is freely available. The treebank is affiliated to UD (Universal Dependencies), project (with 3,700 sentences illustrating Contemporary Standard Romanian, and with 1,200 sentences illustrating Non-standard Romanian). However, the UD annotation system is simpler than ours, and the affiliation of our treebank is possible only with loss of information. We aim to establish an original system of semantic annotation exploiting all the semantic information contained in our treebank (i.e. in the syntactic categories, in the morphological analysis, in the lexical definition of some words, and in the punctuation). We developed some logical structures similar to the AMR (Abstract Meaning Representation) system, but we intend to maintain the form of the Functional Dependency Grammar (FDG) trees for the semantic layer, in order to preserve the isomorphism with the syntactic one. The chosen solution will be justified and compared with other systems.

Keywords: Dependency treebank, non-standard Romanian, logic-semantic layer of annotation, correspondences syntactic-semantics, similarities with other international systems.

1 Introduction

1.1 Perspectives on Developing UAIC-RoDiaTb

The UAIC Dependency Treebank[1] has become an important corpus for Romanian language, with rich morphologic and syntactic information. This treebank is balanced

[1] Software | UAIC NLP (Natural Language Processing) Group, UAIC-RoDia = ISLRN 156-635-615-024-0.

and attempts to illustrate all the styles of the language; the average is 19.29 words per sentence. We consider that the purpose of Natural Language Processing (NLP) is to model the complexity of the human language, and not only to model the man – computer communication in a simplified way.

Although we have made the transposition table for automatically transposing our conventions into UD ones, part of the UAIC-RoDiaTb was transposed in the UD by the RACAI group (Research Institute of Artificial Intelligence), interested only in Contemporary Standard Romanian.

There are many theoretical problems that differentiate us from UD; for example, the treatment of relational words. The syntactic categories are classified according to the UD conventions in what concerns the morphological classes (i.e., adjectival, adverbial, nominal modifier); additionally, we consider that the syntactic information should be correlated with the semantic one.

The semantic richness of our tags derives from the UAIC annotation conventions that are not modern, but classically syntactic, containing 14 different circumstances carefully checked. The challenge is to build mechanisms for the automatic recognition of these 14 values when they are not determining verbs (i.e., a classification as local, temporal, causal determiner of a noun would be useful). We do not claim the perfection of the syntax-semantics isomorphism; however, we watch it closely in order to discover and to surpass its limitations.

In order to preserve all the information which has been automatically annotated and carefully supervised in all the 18,630 sentences, we propose a project that aims to transform the classical treebank into a semantic layer. The similarity of the syntactic and semantic functions has been widely discussed. Fillmore [8], focusing on the transformational grammar theory, has described the deep structure as a semantic one, with categories such as: Agent, Instrument, Objective, Dative, Locative, and Factitive. In his conception, the syntax is the surface structure. Both structures have a single position for each argument.

This conception, underpinning the Semantic Roles of PropBank or FrameNet, is largely used by the computational linguists. However, Fillmore's deep structure has very few semantic categories for the purpose presented above, i.e. only the obligatory verbal dependencies. The modifiers (being mandatory for some verbs) have also semantic roles, the functional words, the markers of tenses, modes, and diathesis, the articles and other morphological categories, and also the punctuation has semantic functions that can be used by other applications such as sentiment analysis, information retrieval, question answering, or temporal structuring of the discourse. This research can be an experiment on Romanian with possible multilingual value.

The syntactic categories are more abstract than the semantic ones; the latter refer to the "deep structure", only in the sense that they are nearest to the communicative purpose. We consider the syntactic and the semantic structures as parallel, since the FDG [14, 20] rejects the concept of deep structure.

For this purpose, we built a new working interface, called Treebank Annotator [11], that allows viewing and comparing two trees of the same sentence in different annotation conventions, and working alternatively with them, see Fig. 1.

Half of the syntactic tags can be automatically replaced with semantic ones. The logico-semantic- argumentative system chosen is able to preserve the annotated and

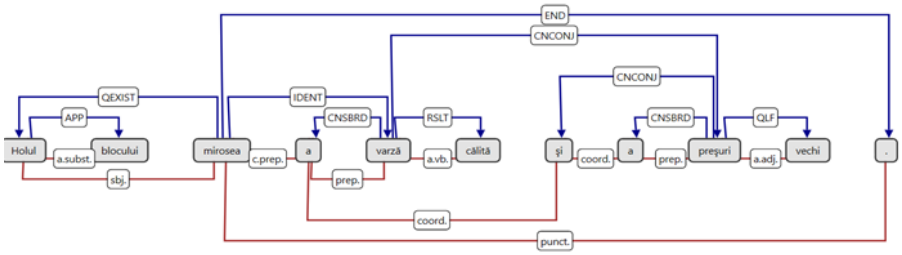


Fig. 1. Comparison between the syntactic and the semantic annotation (The hallway of the block smelled of tanned cabbage and old pretzels).

supervised information of the UAIC-RoDiaTb[2]. The paper also contains a theoretical justification of the system described and an attempt to find future affiliations to international compatible systems. The system is compared with similar works and other solutions will be discussed.

1.2 UAIC-RoDiaTb Syntactic System of Annotation

The UAIC-RoDiaTb is a syntactic treebank based on the FDG grammar. In FDG, the connection is a functional binary relationship between a regent and a dependent. By using this model, the text processing is uniform; any element of the structure, word or punctuation mark, is a node of the tree, in relation to another node.

This kind of treebank is currently being developed for more and more languages. All these corpora respect the FDG rules; however, there are multiple differences between their annotation systems. The coordination is quite difficult to express in this theoretical model, in which the relationship of equality, (horizontal) is not allowed. In our system, the coordination is asymmetrically rendered; the coordinator element is subordinated of the first coordinated and regent for the second.

Another problem with divergent solutions in FDG is the annotation of the relational words. In the Penn treebank, these words are subordinated of the full-meaning words or of the head of clauses that they introduce in the tree. This solution is taken over by many corpora and by the UD system. However, recently, researchers have begun to wonder if this solution does not have more disadvantages than advantages.

(This was one of the proposed themes for the researchers at the Depling 2015 Conference)[3]. In the UAIC-RoDiaTb, the relational words are connectors between the head and the subordinate word, i.e. regents for the word they introduce. The first arguments for this decision are:

- The relational words are marks for the subordination or coordination. The prepositions also impose of the subordinated word the determined or undetermined form, and the case,
- The number of the functional words is small since this lexical category is closed; each of them can be described, they have a number of relations that it can

[2] RoDia is the Romanian word for pomegranate; read on syllables means Romanian Diacronic Treebank.

[3] http://depling.org/depling2015/.

establish, and some formal restrictions imposed of the subordinate. It is possible to formulate a system of rules for a syntactic parser or for a machine learning hybrid or rule-based, given the properties of such relational words.

In our system, the subordinator elements are annotated consequently with the coordinators ones, as subordinated of the regent and regent for the subordinated word or clauses that they introduced.

The syntactic tags of the UAIC-RoDiaTb are classical; for example, there are 14 types of circumstantial modifiers. In fact, the content of this classification of verbal modifiers is more semantic than syntactic, so it contains precious information.

The UD system is based on morphologic information because the morphology part has already been correctly annotated, and the syntactic parsers work better when they make use of this type of information. However, there is redundancy in the UD system that marks twice the morphological information (in the POS-tag and in the dependency tag). Our semantic classification of modifiers is extra information that can be used by future applications.

2 Related Work

2.1 Related Work for Romanian

In the UAIC NLP group, Trandabăț [21] has imported about 1,000 sentences from the English FrameNet. She has translated in Romanian the sentences and has retained their semantic annotation from the English FrameNet. In this way, she has made a first set of semantic annotations on Romanian sentences. Like the English FrameNet, these annotations only cover the core structure of the sentence called Semantic Frame, the (mandatory) predicate arguments, called Semantic Roles, and the semantic function of others members of the structure is not analyzed.

Another research made the classification of verbs like the English VerbNet. The classes of verbs are carefully inventoried for English [13]. The participants at Eurolan 2013 Summer School tried to find some corresponding examples for such classes in eDTLR[4], the electronic transposition of the paper Thesaurus Dictionary [1], to illustrate each class of the Romanian VerbNet [6].

Therefore, in Romanian the most frequent patterns are different and with specific structures by rapport to English. More recent research has shown that there are other parts of speech that can also be logical predicates and can have the same structures of arguments, especially the nouns or adjectives derived from verbs, and the semantic roles must be extended to these other heads (see below).

2.2 Diversity of Semantic Approaches

There is no universal consensus about semantic annotation, and the number of semantic categories is also disputed. Many papers propose a small number of semantic relations

[4] eDTLR was built during 2007-2010 by a group of computer scientists and linguists from three Academic Linguistic Institutes from Iasi, Cluj, and Bucharest, and from two Artificial Intelligence Academic Institutes.

[9-10]. Amaro [2] describes a more extensive corpus work: she analyses a number of 26 relationships from the Portuguese WordNet and studies them in 35,000 contexts, with the aim of creating a semantic annotation based on lexical and syntactic information. In another paper [5], the authors have the purpose of developing semantic annotations for the PAS (Predicate Argument Structure) in the PropBank [17]. They exemplify some semantic tags used by English Vallex, (e.g., ACT (Actor), PAT (Patient), ADDR (Addressee), ORIG (Origin) and EFF (Effect), considering them as being too descriptive for their purpose.

Previously, the annotation effort has focused on event relations expressed solely by verbs, but the meaning of words is not necessarily linked to their morphological value - nouns, adverbs, and interjections can also express an event. They consider it necessary to expand the PropBank annotations so as to provide coverage for nouns, adjectives, and complex predicates. This research is called Predicate Unification.

The FrameNet annotations for these various logical predicates split them in different frames. For example, fear-noun fall into the 'Fear' frame, fear-verb falls into the 'Experiencer Focus' frame, and afraid-adjective is included in both. As a result, sentences describing the same eventuality would not be recognized as synonymous under the FrameNet annotation.

On the contrary, Ștef[nescu [19], study the semantic similarity experiencing latent semantic analysis models on two large corpora, Wikipedia and TASA (Touchstone Applied Science Associates). The clustering model investigates the similarity between words without proposing any ontology (tags for syntactic relationships). In our NLP group, a similar research is RoPAAS (Romanian Predicate Argument and Adjunct Structure), see [18]. The semantic logic Romanian approaches also consider that the logical predicates can be expressed by adjectives, interjections and adverbs [24].

2.3 Comparison between UAIC Semantic Ontology and the Tectogrammatic Layer of the Prague Dependency Treebank (PDT)

The PDT[5] is a long-term project, started in 1996. In 2003, in [4], the Czech researchers described them as a three-level annotated corpus of 1.8 mil. tokens. The first level is the morphological annotation; the second is the superficial syntactic annotation, affiliated to UD, and the third one is called the tectogrammatical level, or the level of linguistic meaning (based on the framework of Functional Generative Description).

The UAIC-RoDiaTb also has a first morphological level, but it is included in the syntactic level. The superficial syntactic layer of PDT has less information than the old classical syntactic layer of the UAIC-RoDiaTb, and the verbal modifiers are not classified. The tectogrammatic layer of PDT is obtained after some transformations. There are a big number of relations called "functions" abbreviated "func":

/ACT/PAT/ADDR/EFF/ORIG/ACMP/ADVS/AIM/APP/APPS/ATT/BEN/CAUS/CNCS/COND/CONJ/COMPL/CPR/CRIT/CSQ/CTERF/DENOT/DES/DIFF/DIR1/DIR2/DIR3/DISJ/ETHD/EXT/FRWH/GRAD/ID/INTF/INTT/HER/LOC/MANN/MAT/MEANS/MOD/NORM/PAR/PREC/REAS/REG/RESL/RESTR/RHEM/RSTR/SUBS/TFHL/THL/THO/TOWH/TPAR/TSIN/TTILL/TWHEN/VOC/VOCAT/NA/SENT/

[5]http://ufal.ms.mff.cuni.cz/pdt/pdt.html

The information is organized in more attributes, each of its values, while in our system, all semantic information is encoded in the deprel attribute: modifiers are classified in a similar way as in the classical syntactic convention of UAIC- RoDepTb and in the ontology proposed below: CNCS = Concession, CAUS = Causative, CSQ = Consequence, ADDR = Addressee, COND = Conditional, etc.; moreover, an automatic transposition of the UAIC semantic system of annotation in the PDT system will be possible.

In a report published on PDT site [16], the recent modifications and the direction of the development of this large resource have been described. The PDT texts are in the journalistic style, illustrating the contemporary language.

The complexity of the information annotated in the third layer of the PDT allows for many different lines of research.

A preoccupation for functional words, and the intention of annotating the meaning of these words is common for PDT.02 and the ontology presented below; i.e., the meaning of grammatical categories that these words form can be semantically annotated as past, passive, reflexive, reciprocal, continuous, etc.

Different sections of the report describe the attributes *grammatemes: typgroup, factmod, diatgram.* The *typgroup* aims at refining the category of singular or plural, the *factmod* is conceived to annotate the meaning of the mode of the verb, which expresses a real action, or a possible, uncertain, claimed action.

In the ontology proposed below, there are the following categories: Past, Future, Continuous for annotating the *grammatemes*, and the categories Generic, Uncertain, Imperative, and Optative for the *factmod*.

The attribute *diatgram* formalizes the meaning of the verb diathesis (voices): Passive, Reciprocal in our system. For the PDT Refl 1 and 2, UAIC have the tags: Continuative, Dynamic, and Impersonal.

The values of *sentmod* attribute are: *enunc, excl, dezid, imper, inter,* [22] which annotate the type of the sentence, from the pragmatic perspective of the sentence emitter. The system that we present has also the categories Interrogative, Imperative, Exclamation.

The development of this important resource, PDT, especially after the introduction of the tectogrammatic layer, shows how vast the prospects that open the semantic annotation for the future reuse of the corpus are.

2.4 Comparison between UAIC Semantic Ontology and the Abstract Meaning Representation (AMR) Project

In this paper, „ontology" refers to the set of semantic-logic concepts chosen as tags for the semantic annotation, and to the attribution of these tags to syntactic relations, to morphologic categories, or to particular words.

AMR is a semantic representation language proposed by Bănărescu [3] that uses graph notations for computer processing and a modified form of the PENN annotation [11] for human reading and writing. AMR graphs are rooted, labeled, directed, acyclic (DAGs); they are able to represent various linguistic phenomena, such as semantic roles, co-reference, questions, modals and negation, named entities, copula, reification, and so on [23]. The nodes of an AMR graph are labeled with concepts while the edges are labeled with relations.

Similarly, to the UAIC ontology, AMR provides full sentence deep semantic representations, not only the mandatory relations. AMR annotates sentences independent of context (i.e. it takes the sentence, and not the text as the unit of annotation). Even if some discourse relations, such as contrast ('but') and concession ('even though'), and co-reference are already represented in the AMR annotations at intra-sentential level, the future directions for AMR imply new possibilities for representing the inter-sentential co-reference and discourse relations.

Another similarity is the use of a relatively large number of semantic relations. AMR uses approximately 100 relations, (the UAIC system has 96) that include frame arguments adopted from the PropBank annotations in OntoNotes [12], and other semantic relations. The paper [3] give a comprehensive list of these relations:

- **General semantic relations**: :accompanier, :age, :beneficiary, :cause, :concession, :condition, :consist-of, :degree, destination, direction, domain, duration, etc.,
- **Relations for quantities**: :quant, :unit, :scale,
- **Relations for date-entities**: :day, :month, :year, etc.

The difference between AMR and the UAIC system is that the first one does not annotate word tokens in a sentence, but concepts. This means that AMR generalizes over morpho-syntactic idiosyncrasies such as word category, word order, or morphological variation. Consequently, content words are annotated as concepts (i.e. they drop such information as plurality, articles, or tense and aspect) and they can correspond either to predicate-like elements (e.g.: 'teach', 'teacher', 'attractive', 'acquainted'), or to special (English) keywords (e.g.: 'person', 'name', special entity-types 'distance-quantity' and logical conjunctions 'and', or', etc.). The function words are either annotated by means of the semantic relations they represent, or omitted if they do not contribute to the meaning of a sentence.

AMR differs from other ontologies since it combines multiple layers of linguistic annotation in a single structure with the aim of obtaining a high degree of generalization, both at one-language level and cross-linguistically. Keeping a single structure rather than multiple layers is also correlated with losing a lot of language-specific information since AMR is generally considered to 'abstract away' from the morphological and syntactic variations that are present in a language, and this accounts for many of the cross-lingual differences.

Although this approach leads to one of the AMR strengths, which is the ability to encode in a single representation multiple sentences sharing the same meaning even if not identically worded, it also results in some 'side-effect' limitations. These refer, for example, to the fact that AMR does not distinguish between real events and hypothetical, future, or imagined ones since it does not encode tense and aspect features because they do not generalize well cross-linguistically [3, 23].

However, the AMR quality of collapsing more ways of saying things made it interesting for MT (Machine Translations) experiments, in order to see if this representation can serve, e.g., as a useful, minimally divergent transfer layer in machine translation [26].

Table 1. The syntactic and semantic roles depending on the type of judgment.

Type of judgment	SSubject	Direct object	Predicative noun	Ot **Other**
Process	Agent	Result	none	none
Performance	Performer	none	Qualifier	Circonstances
Actantial	Agent	Patient	none	none
Experience	Experiencer	Experience	none	Circonstances
Existence	Existent	none	none	Circonstances
Communicative	Emitter	Content	none	Recipient
Definition	Definiens	none	Definiendum	copula
Identity changing	Definiens	none	Definiendum	copula
Possession	Possessor	Posseded	none	none
Characterization	Theme, content	none	Qualifier	copula

3 Mapping the Classical Syntactic Relationships onto the Semantic Categories Proposed

3.4 Monosemantic Syntactic Tags

In the UAIC-RoDiaTb there are 44 syntactic tags, 20 of which having a unique translation into semantic tags: superl. (superlative), comp. (comparative), ap. (apposition), incid. (incident), neg. (negation), voc. (vocative), c.ag. (agent complement), and 13 circumstantial modifiers, except the modal one, which can have more values.

3.5 Categories Dependent on the Morphological Tag or on the Word Form

There are categories which can be strictly separated considering the meaning of their morphological classification, i.e. we can formulate rules for the correspondence of syntactic and morphologic annotation with semantic tags. Using this information, the syntactic deprels can be automatically changed in semantic ones by rules with two or more conditions. Examples:

- The types of articles, annotated det., can have the following semantic values: cel, cea, cei, cele, etc. (En: *the* + adjective)= Deictic; un, o, niște, etc. (En: *a*) = Undefined; al, a, ai, ale, etc., (En: *of the*) = Possessive,
- For aux. (auxiliary), the occurrences can have the following semantic values (in agreement with the meaning of verbal forms obtained with these auxiliaries): Optative, Future, Past, Passive. For the auxiliary putea (En: *can*) the semantic values are: Potentiality, Ability, or Competence,
- Frequently, the subordination marks, prepositions, conjunctions, or Rw, Dw, Pw are marks of the semantic values: pentru (for) has the value Purpose; fiindcă,

deoarece, căci (because) has the value Causative; deși, măcar că, (although) has the value Concession; dacă (if) has the value Condition, etc,

- The syntactic-morphological tag a.pron. can have the following semantic values: Possessive, Deictic, Negative, Interrogative, Emphatic, Undefined or Quantifier: universal for: toți, fiecare, oricare, (all, any, every),
- The tag punct. can have the following semantic values: Exogene, Unnecessary, Dislocation, Connect-reunion, End, Exclamation, Interrogation. In Druguș [7] the punctuation elements are considered as linguistic connectors, having logical and semantic values; we have already developed this theory by interpreting semantically each punctuation element, which, in FDG, must be annotated as part of the syntactic or semantic tree.

3.6 Syntactic Tags Semantically Polyvalent

The polyvalent relations are: a.adj, a.adv., a.subst., a.vb., c.d., c.i., c.prep., a.subst., c.c.m., sbj., n.pred., el.pred. For the sbj. and n.pred./el.pred., c.d./c.i. (subject, predicative noun/element, direct /indirect object); there is a reduced number of possible values, depending on the type of judgment (see Section **4.1.**).

However, the a.vb. (verbal attribute), a.subst. (attribute expressed by noun), and c.prep. (prepositional object) are syntactic tags without semantic value, established according to formal morphological criteria. They can have almost every semantic value.

4 Types of Judgments and Connectors

4.4 Judgments

The judgments contained in a sentence are not focused only on an event. We have proposed below a classification of sentences including not only events, but also definitions, descriptions, speech acts, or existential affirmations. For each type of judgment, the roles are different and we can use this information for the detection of roles by a semantic parser hybrid or rule based. The roles for each type of judgment are shown in Table 1.

4.5 Connectors

We can consider a many words and punctuation elements as logical connectors. Emanuel Vasiliu has devoted most of his books to the relationship between logical artificial languages and natural language studies, either in the clause or in the sentence [24-25], especially regarding the translation of logical connectors and categories into natural language words. The translations of the logical connectors into the natural language are: ~ = (negation) ”not”; U = (reunion) și “and”; ∩= (disjunction) = sau ”or”; ⊃ = (implication) = deci ”so”.

In terms of logics, they form horizontal expressions, and in terms of the dependency UAIC convention, there are oblique descending lines, the connector being posted between the connected words, i.e. subordinate of the first and regent for the second.

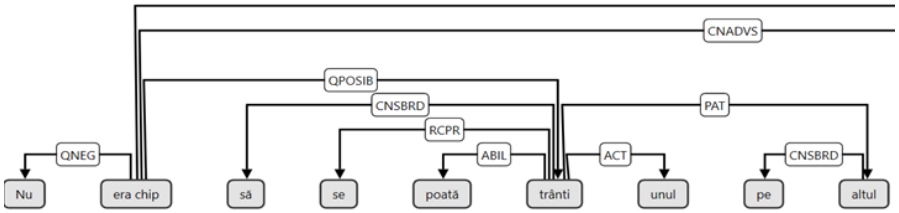


Fig. 2. The operator possibility: “There was no way they could prevail against one another”.

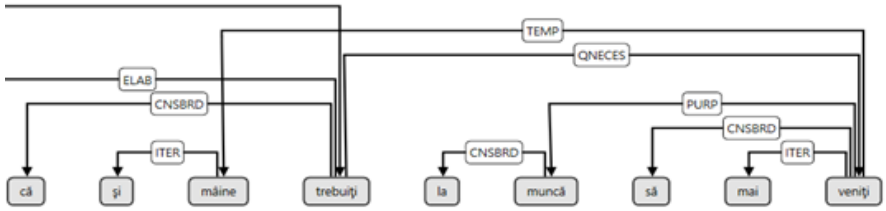


Fig. 3. The operator necessity: “that tomorrow you must come to work longer”.

The binary connectors will be annotate as: Connect:reunion, Connect:opposition, Connect:disjunction; Connect: subordination.

The connectors: ~ = “not”; ∀= “all” are not dyadic, but monadic. We decide to call it Quantifiers, because they do not connect two elements. The Quantifier:negation is subordinated of the word which is negated.

The Quantifier:universal is subordinate of a noun. The modal connectors, also monadic, consequently annotated as Quantifier:necessity (the symbol = □) and Quantifier:possibility (the symbol = ◊) are expressed in natural language by a sentence head that must have a subordinate subjective clause. Examples: Trebuie (să) = □, este probabil (că) = ◊. “We need (to ...), It is likely (that...).”. The modal quantifiers will be considered as regents of the clause which they modalise, see Fig. 2 and Fig. 3.

5 Conclusions

This paper proposes a type of semantic annotation with more categories since we aim to keep all the information that has been annotated in the classical syntactic layer; as we have argued in this paper, this information is important since it can be exploited by other applications. Another purpose was to find an international annotation with similar categories in view of a future affiliation.

The similarities with the tectogrammatic layer of PDT and with the AMR logical categories are obvious. However, there are also differences since the resultant graph of the AMR semantic annotation is not a dependency tree, and the nodes are not words, but concepts. In order to show the isomorphism between the syntactic and the semantic structures, we chose to build a corpus of semantic dependency trees, which is similar to the tectogrammatic layer of the PDT.

The set of annotations has been successfully experimented. The UAIC treebank has a parallel corpus in semantic format, having now 5,500 sentences. Moreover, in a

collection of 400 sentences, the syntactic relations with a high degree of ambiguity have been manually annotated by three experts, and their agreement has been of 85% (with the same solutions adopted and without different annotation for similar situations).

References

1. Academy of Romania: Thesaurus-Dictionary of Romanian Language. Compound by 2 series: I: 1913–1949. Dictionary of Romanian Language. Socec, Universul Publishing, Bucharest, and II: 1965–2010. Dictionary of Romanian Language. Romanian Academy Publishing, Bucharest (1913–2010)
2. Amaro, R.: Extracting semantic relations from portuguese corpora using lexical-syntactic patterns. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pp. 3001–3005 (2014)
3. Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., Schneider, N.: Abstract meaning representation for sembanking. In: Proceedings of the 7th linguistic annotation workshop and interoperability with discourse, pp. 178–186 (2013)
4. Böhmová, A., Hajič, J., Hajičová, E., Hladká, B.: The prague dependency treebank: A three-level annotation scenario. Text, Speech and Language Technology, Springer, vol. 20, pp. 103–127 (2003) doi: 10.1007/978-94-010-0201-1_7
5. Bonial, C., Bonn, J., Conger, K., Hwang, J. D., Palmer, M.: PropBank: Semantics of new predicate types. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pp. 3013–3019 (2014)
6. Cristea, D., Mihăilă, C., Forăscu, C., Trandabăț, D., Husarciuc, M., Haja, G., Postolache, O.: Mapping Princeton wordnet synsets onto Romanian wordnet synsets. Romanian Journal on Information Science and Technology, vol. 7, no. 1–2, pp. 125–145 (2004)
7. Fillmore, Ch. J.: The case for case. Universals in linguistic theory, (Part Two). Bach, E. and Harms, R. T. (eds), Holt, Rinehart and Winston Publishing, pp. 1–25 (1968)
8. Fukuda, S., Nanba, H., Takezawa, T.: Extraction and visualization of technical trend information from research papers and patents. D-Lib magazine, vol. 18, no. 7–8 (2012) doi: 10.1045/july2012-fukuda
9. Gupta, S., Manning, C.: Analyzing the dynamics of research by extracting key aspects of scientific papers. In: Proceedings of 5th International Joint Conference on Natural Language Processing, pp. 1–9 (2011)
10. Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., Weischedel, R.: OntoNotes: The 90% solution. In: Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers, pp. 57–60 (2006)
11. Levin, B.: English verb class and alternations. A Preliminary Investigation. University of Chicago Press (1993)
12. Mel'Cuk, I. A.: Dependency syntax: Theory and practice, Buffalo, Suny Press (1988)
13. Matthiessen, C. M. I. M., Bateman, J. A.: Text Generation and systemic-functional linguistics: Experiences from english and japanese. Printer Publishers (2002)
14. Mikulová, M., Bejček, E., Mírovský, J, Nedoluzhko, A., Panevová, J., Poláková, L., Stranák, P., Ševčíková, M., Žabokrtský, Z.: From PDT 2.0 to PDT 3.0 (modifications and complements), ÚFAL technical report (2013) https://ufal.mff.cuni.cz/techrep/tr54.pdf
15. Palmer, M., Gildea, D., Kingsbury, P.: The proposition bank: An annotated corpus of semantic roles. Association for Computational Linguistics, vol. 31, no. 1, pp. 71–105 (2005)
16. Perez, C. A., Mărănduc, C., Simionescu, R.: Ro-PAAS a resource linked to our UAIC-Ro-Dep-treebank. Advances in Artificial Intelligence and Soft Computing, In: Proceedings of 14th Mexican International Conference on Artificial Intelligence. Springer, vol. 9413, pp 29–46 (2015) doi.org/10.1007/978-3-319-27060-9_3

17. Ștefănescu, D., Banjade, R., Rus, V.: Latent semantic analysis models on Wikipedia and TASA. In: Proceedings of Ninth International Conference on Language Resources and Evaluation, pp. 1417–1422 (2014)
18. Tapanainen, P., Jarvinen, T.: A non-projective dependency parser. In: Proceedings of the 5th Conference on Applied Natural Language Processing, pp. 64–71 (1997)
19. Trandabaț, D.: Natural language processing using semantic frames. PHD Thesis, Faculty of Computer Science, Al. I. Cuza University, Iași (2010)
20. Urešova, Z.: Building the PDT-Vallex valency lexicon. In: Proceedings of the Corpus Linguistics Conference (2009)
21. Vanderwende, L., Menezes, A., Quirk, Ch.: An AMR parser for english, french, german, spanish and japanese and a new AMR-annotated corpus. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pp. 26–30 (2015) doi: 10.3115/v1/N15-3006
22. Xue, N., Bojar, O., Hajic, J., Palmer, M., Uresova, Z., Zhang, X.: Not an interlingua, but close. Comparison of English AMRs to chinese and czech. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pp. 1765–1772 (2014)

ISSN 1870-4069

Corpus-based Automatic Text Expansion

Balaji Vasan Srinivasan[1], Rishiraj Saha Roy[2], Harsh Jhamtani[3],
Natwar Modani[1], Niyati Chhaya[1]

[1] Adobe Research Big Data Experience Lab,
Bangalore,
India

[2] Max Planck Institute for Informatics,
Saarland Informatics Campus,
Germany

[3] Carnegie Mellon University,
Language Technology Institute,
USA

{balsrini, nmodani, nchhaya}@adobe.com,
rishiraj@mpi-inf.mpg.de, jharsh@cs.cmu.edu

Abstract. The task of algorithmically expanding a textual content based on an existing corpus can aid in efficient authoring and is feasible if the desired additional materials are already present in the corpus. We propose an algorithm that automatically expands a piece of text, by identifying paragraphs from the repository as candidates for augmentation to the original content. The proposed method involves: extracting the keywords, searching the corpus, selecting and ranking relevant textual units while maintaining diversity in the overall information in the expanded content, and finally concatenating the selected text units. We propose metrics to evaluate the expanded content for diversity and relevance, and compare them against manual annotations. Results indicate viability of the proposed approach.

Keywords: Corpus, text expansion, automated text.

1 Introduction

While automated text summarization has been thoroughly researched over the last decade, the reverse task of "expanding" a piece of text has not been explored widely. Our work in this paper is motivated by the use case of automatically "resizing" textual content according to its delivery channel which can be assisted via algorithmic text expansion.

While channels like social media require content limited to a few characters, channels like websites, blogs or emails require an elaborate version of the same content. Content authors in an organization are under severe time pressure to deliver such modified versions along with numerous other stylistic personalizations of the same

piece of content. We believe that automating text expansion can be an important step towards accelerating the authoring workflow for textual content.

In this work, we propose algorithms that take a piece of textual content and expand it to a desired size by adding required content from a repository. The input is a short snippet composed by the author. A search query is constructed using the representative terms in the snippet, and is used to fetch relevant content from the corpus. We propose two algorithms to choose desired content from the retrieved list to produce the final expansion, and evaluate their performance on a real-world data set.

The rest of the paper is organized as follows. We differentiate existing work in this space from our current problem in Section 2. We introduce the proposed algorithm in Section 3 and evaluate its performance based on human annotations in Section 4. We propose metrics to evaluate the expanded content and correlate them with human annotations. Finally, we evaluate the expansion algorithm on a larger dataset to show the viability of the proposed algorithm in Section 5. Section 6 concludes the paper.

2 Related Work

Mihalcea et al. [5] identify key concepts in a document and link them to corresponding Wikipedia pages. While this helps to identify relevant Wikipedia areas for the document, this will not help in expanding the seed content from these input sources.

Li et al. [4] use a language model and a *topic development curve* to identify the most relevant text snippet in a document corresponding to a query. Snippets consist of phrases or sentences from multiple parts of the document, without sufficient context. While snippets may be useful in elegant presentation of a search engine results page, they are not suitable for text expansion.

Schlaefer et al. [8] aim to enhance question-answering by expanding a 'seed document' using web resources. The most relevant paragraphs (nuggets) are combined to provide the answers. While avoiding lexical redundancy, the authors retain semantic redundancy as it is desirable to enhance question-answering performance. This may however not be ideal for a content author as (s)he would want to avoid any type of content redundancy for human consumption, be it lexical or semantic. In the proposed algorithms, we address this by jointly optimizing for relevance and diversity of the expanded content.

Taneva and Weikum [9] identify relevant text snippets ('gems') by using an integer linear program to maximize relevance of selected words, and prefer the selection of contiguous words. However, such a method can result in only fragments of a sentence being selected, since the linear program is formulated at a word-level, thereby affecting readability. Biases may also be introduced in the expanded content which may not be preferable to an author.

There is a significant body of work in the domain of text summarization [7]. Text expansion could be perceived as a summarization task once we have identified the required candidate paragraphs from the repository. However, lack of notions of information coverage (maintaining the distribution of concepts between input and summary) in expansion makes it different from summarization.

3 Text Expansion

The primary requirements in expanding a content are that the resulting text should be relevant to what the author is building and also be diverse in the overall information present. We aim to achieve both these requirements with the proposed framework.

The input to our algorithm is a *content snippet* that the author is looking to expand, and the desired *length of the target* expansion (in words). The first step in our algorithm is extracting the top-k keywords in the snippet using the inverse document frequency (IDF) of the words in the corpus, thus capturing the most significant keywords in the snippet with respect to the corpus. A query q is then constructed by concatenating these k keywords.

The choice of k determines the relevance and the amount of content that is available and fetched from the repository. A lower value of k results in the query under-representing the content and fetching articles that may not be very relevant. On the other hand, a higher value of k will result in a very specific query that might not fetch many results from the repository.

We use q to retrieve indexed content from the corpus. The retrieved content is split into paragraphs $\{P_1 \dots P_n\}$; which are the candidates for inclusion in the expanded content. Paragraphs are preferred over sentences because they are more likely to preserve (local) coherence in the final text. We assign every paragraph with a relevance score to the query based on a Lucene index. Often, paragraphs with high relevance scores contain significant information overlap (and hence redundancy). Therefore it is important to choose the relevant paragraphs but still account for the diversity in the overall material. We propose two approaches for this below.

3.1 Maximal Marginal Relevance (MMR)-based Ranking

Our first algorithm is inspired from **Maximum Marginal Relevance** (MMR) [1] for selecting the candidate paragraphs. MMR is used for obtaining diversified search result ranking often with multiple optimization objectives, e.g., relevance and diversity. At each iteration of the MMR algorithm, the best item is selected from the set of candidates by minimizing a cost function. For expansion, the cost function is formulated as:

$$\max_{P_i \in R \setminus S} \left[(\lambda \times score_{rel}(q, P_i)) - ((1-\lambda) \times \max_{P_j \in S} (sim(P_i, P_j))) \right], \qquad (1)$$

where, R is the set of all candidate paragraphs, S is the subset of R that is already selected for the expansion, $R \setminus S$ represents the set of unselected paragraphs so far, $score_{rel}$ is the relevance score of paragraph P_i w.r.t query q, $sim(P_i, P_j)$ is the similarity (e.g., cosine similarity) between the vector representations of paragraphs P_i and P_j (reflecting the degree of content overlap), and $\lambda \in [0, 1]$ is a tunable parameter that reflects the trade-off between relevance and redundancy.

At each step, the paragraph that maximizes the above cost function is added to the expanded content S and continued till the length of the expanded content reaches the desired limit, or the list of candidates is exhausted.

3.2 Graph-based Ranking

Our second algorithm is based on the graph-based ranking in [6]. We represent each paragraph P_i as a node $v_i \in V$ in a weighted graph $\mathcal{G} = (V, E, W)$. We assign an initial "reward" r_i^0 for P_i as the relevance of the paragraph P_i to the query q. The cost c_i of the paragraph P_i is taken as the number of words in P_i.

An edge $e \in E$ between vertices v_i and v_j exists if there is a non-zero similarity between P_i and P_j, and is weighted by a similarity function (again, like cosine similarity) w_{ij} under a vector space representation. The gain G_{v_i} of including a node v_i in the expanded content at iteration l is defined as its current discounted reward plus the weighted sum of the current discounted rewards of all immediate neighbors (N_i) of v_i in $\mathcal{G}$, given by:

$$G_{v_i}^l = r_i^{l-1} + \sum_{v_j \in N_i} r_j^{l-1} \times w_{ij}. \quad (2)$$

At step l, we add v_i^* with cost c_i less than the remaining budget and maximum gain-to-cost ratio G_{v_i}/c_i to our expansion. The rewards of the neighbor nodes v_j of v_i^* are then updated as $r_j^{l+1} = r_j^l \times (1 - w_{i^*j})$. This avoids inclusion of similar paragraphs thus ensuring diversity. We stop when there are no nodes left with cost lower than the available budget.

4 Experimental Evaluation

For our initial evaluation, we used a repository of 215 articles (indexed via Apache Lucene) from a proprietary forum including articles around key product features and troubleshooting instructions. We extracted 30 short text fragments and applied the proposed approaches to expand the original snippets using the repository. The input snippets had 33.9 words on an average, ranging from 4 to 86 words. The expansions were run with a target length of 500 words, with $k = 10$. The two methods generated a total of $30 \times 2 = 60$ expansions.

4.1 Human Evaluation

We obtained scores from 30 human annotators, each annotating 4 of the generated expansions, evaluating the *relevance* of the expanded content to the seed content and its content *diversity*, on a scale of $0 - 7$. We collected 120 annotations, each of the 60 expansions being rated twice while ensuring that the same annotator does not annotate the output from both algorithms. Fig. 1 plots the fraction of times (y) an expansion received a score of at least x computed based on the cumulative distribution of the scores from the kernel density estimates.

The two approaches are comparable on relevance, as the same keyword extraction and search process applies for both of them. Diversity was observed to be better for MMR, possibly because it directly optimizes for low content-level overlap in its objective function via the choice of λ.

4.2 Automated Evaluation

While our results with human annotations are encouraging, evaluating the expansion performance on larger datasets requires a metric-based objective estimate of relevance and diversity that correlates well with human annotations. While metrics like KL-divergence [7] are widely used for measuring summarization quality, they cannot be applied as they are for evaluating expansion, because of the differences in the underlying tasks. We therefore propose two metrics to capture the degrees of relevance and diversity in the expanded content.

To measure the relevance of the expanded content to the input, we compute the similarity between each paragraph in the expanded content and the input. The average of maximum similarity of every paragraph in the expansion against all input text units can be computed as the relevance, but this will yield higher relevance even when the expanded text units match with very few input text units. On the other hand, taking an average will lead to a reduced relevance score unless it is relevant to *all input* text units. To address these issues, we use a *decayed weighted average* for computing the relevance:

$$\mathrm{rel}(c_{\mathrm{inp}}^{1\ldots N}, c_{\mathrm{exp}}^{1\ldots M}) = \frac{1}{M}\sum_{i=1}^{M}\frac{\sum_{k=1}^{TopK(c_{\mathrm{inp}},c_{\mathrm{exp}}^{i})}\gamma^{k}\,\mathrm{sim}(c_{\mathrm{exp}}^{i}, c_{\mathrm{inp}}^{k})}{\sum_{k=1}^{K}\gamma^{k}}, \tag{3}$$

where $c_{\mathrm{inp}}^{1\ldots N}$ are the text units in the input and $c_{\mathrm{exp}}^{1\ldots M}$ are the text units in the expanded content. $TopK(c_{\mathrm{inp}}, c_{\mathrm{exp}}^{i})$ returns the top-K text units in the input content similar to c_{exp}^{i} (in the decreasing order of their similarly as computed by $sim(c_{\mathrm{exp}}^{i}, c_{\mathrm{inp}}^{k})$). The parameter γ, $(0 \leq \gamma \leq 1)$, penalizes the addition of a text unit that is similar to only a small set of the input text units via a decayed-weighted-average. The $sim()$ could be a standard similarity function between text units. We use cosine similarity here.

The Pearson Correlation Coefficient between the scores from Eq. 3 against the human evaluation is 0.7130 indicating a strong correlation. We also compute the Wilcoxon-Mann-Whitney (WMW) statistic (extended from learning-to-rank problems [2]) between all the scores from the same annotator and the corresponding scores given by Eq. 3 was observed to be 0.9008. The WMW statistic measures the probability that any pair of expanded content samples is ordered correctly based on Eq. 3 against the annotator's ranking.

To compute the diversity within the expanded content, we used a decayed-weighted average of similarity within the text units of the expanded content (similar to the relevance computation). A measure similar to Eq. 3 would give the information overlap (redundancy) within the expanded content, and the diversity is measured by modifying it as:

$$\mathrm{div}(c_{\mathrm{exp}}^{1\ldots M}) = 1 - \frac{1}{M}\sum_{i=1}^{M}\frac{\sum_{k=1}^{TopK(c_{\mathrm{exp}},c_{\mathrm{exp}}^{i})}\gamma^{k}\,\mathrm{sim}(c_{\mathrm{exp}}^{i}, c_{\mathrm{exp}}^{k})}{\sum_{k=1}^{K}\gamma^{k}}. \tag{4}$$

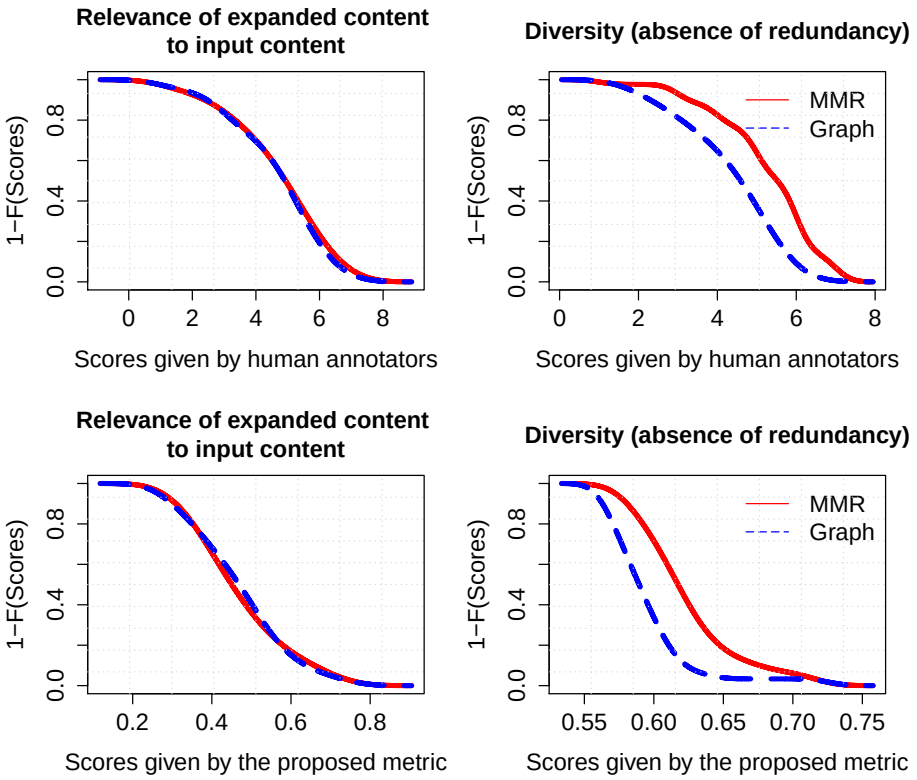


Fig. 1. Evaluation of the two proposed expansion approaches based on annotations of 60 different expansions, from 30 input fragments and two approaches. Each expanded content was annotated by 2 independent annotators. Every point *(x,y)* on the plotted graph indicates the percentage *y* of expanded content that was rated at least *x* by the user. *F* is the cumulative distribution computed using kernel density estimates of the score distribution.

Note that the arguments to the $TopK$ function are different in Eqs. 3 and 4. Eq. 4 captures the similarity within the expanded content and uses it for the diversity computation.

The Pearson Correlation Coefficient between Eq. 4 and the human evaluation is 0.3730, indicating a moderate correlation. The WMW statistic was 0.7795 indicating a good agreement with the human annotators.

Fig. 1 plots the cumulative distribution of relevance and diversity (Eqs. 3 and 4) similar to Fig. 1. Similarity of the two distributions further established a strong correlation between proposed metrics and the human annotations. Note that a relevance score reaches a maximum of 0.75 with a median around 0.5. The cumulative distribution of the diversity is also very similar to that from the annotations, with some deviations as indicated by a lower correlation coefficient. We note that the diversity score reaches a maximum value of 0.7 with a median of around 0.6.

Table 1. A sample input from the Australian legal dataset and the expanded content from the two proposed algorithms (with a desired size of 500).

Input Content Snippet	**Expanded with the MMR-based approach in Sec. 3.1**	**Expanded with the Graph-based approach in Sec. 3.2**
Damages claimed from respondents for breach of guarantee of profit shortfall. First respondent had ostensible authority to bind second respondent to oral variation. Profit shortfall amount for 1998 contracts evidence agency.	However it became clear during cross-examination of Mr Forbes and Mr Brauer that the sales which the respondents claimed should have been credited to the 1998 year actually took place in 1997, and were properly accounted as 1997 sales, as claimed by the applicants. In summary, the respondents claimed that these documents were critical to properly investigating: It was not in contention between the parties that the source financial documents were missing and unavailable. Did Forbes Australia experience a profit shortfall in the financial year ending 31 December 1998?	Did Forbes Australia experience a profit shortfall in the financial year ending 31 December 1998? The material is relevant to both the applicants' claims concerning the 1998 profit shortfall and the respondents' defence. 2. a claim for $1,691,284 which is alleged to be the profit shortfall in respect of the 1999 calendar year. It also follows that the thirty-eighth and thirty-ninth respondents should recover judgment for breach of duty. That shortfall was claimed in the amount of $71,663.65.

5 Experiments on a Public Dataset

Finally, we evaluate our proposed approaches on the Australian Legal Case Reports dataset[1], a collection of 3890 legal cases from the Federal Court of Australia (FCA) from 2006 to 2009. The dataset includes a gold standard summary for every case in the form of 'catchphrases' and 'key sentences' [3]. The legal articles were 6406-word long on an average, while the summaries were 65-word long on an average.

We used the entire set of cases as the content repository and used the gold standard summaries as input to our expansion algorithms. Note that our objective is not to reconstruct the original content from the summary, but rather to test the quality of expansion across several pieces of expanded content.

Table 1 shows a sample input and the corresponding output expanded by the two proposed algorithms. Fig. 2 plots the relevance and diversity of the expanded content for various output sizes based on Eqs. 3 and 4. Fig. 2 also shows the similarity of the expanded content to the original content based on Eq. 3 across the two approaches.

The medians of the relevance and diversity across all the runs are approximately 0.5 and 0.65 respectively, similar to the distribution obtained for the annotated dataset. This indicates a similar quality in the expanded content for the two datasets that we have experimented with. Fig. 2 indicates that the MMR-based algorithm performs marginally better that the graph-based one in terms of relevance and diversity for small expansion sizes.

[1] http://archive.ics.uci.edu/ml/datasets/Legal+Case+Reports, Accessed 16 March 2017.

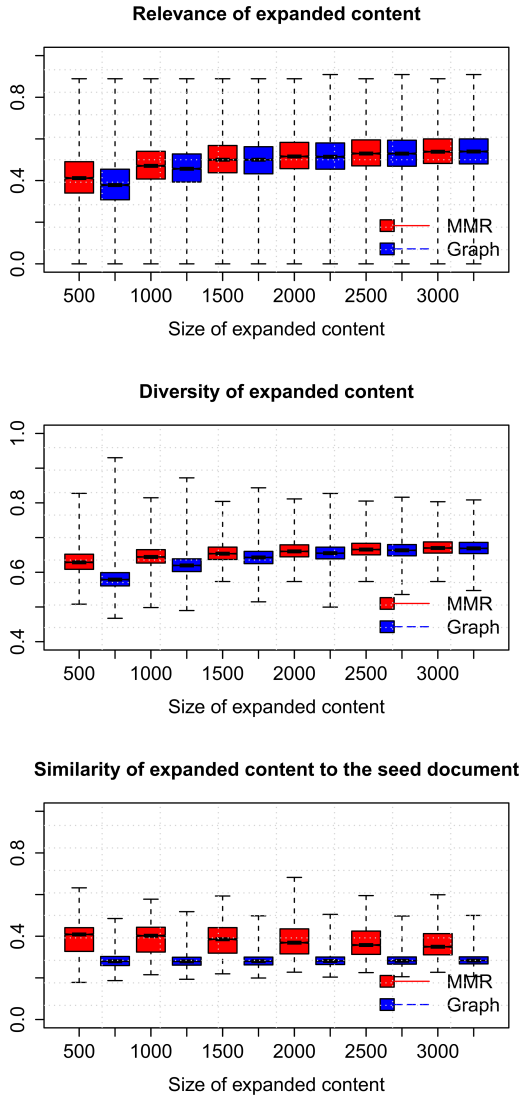


Fig. 2. Relevance and diversity of the expanded content across 3886 samples from the Australian legal dataset for various target sizes.

However, for larger expansion, the relevance and diversity of both MMR and graph-based expansions become comparable. The relevance and diversity that the addition of a new paragraph brings to the expansion, perhaps becomes very low beyond a certain output size leading to eventual saturation of these scores, as seen in Fig. 2.

The proposed approach does not aim at reconstructing the original content whose summary was used for the expansion. However a certain degree of similarity to the original content is desirable from an authoring perspective.

We therefore compute the similarity of the expanded content with the original content using Eq. 3 with $K = 1$. The similarity is higher for the MMR-based approach than the graph-based expansion. For both the approaches, the similarity marginally decreases for higher expansion sizes perhaps because of the algorithms' quest for content diversity.

6 Conclusions and Future Work

We studied the problem of expanding a piece of text by reusing content from an existing corpus and proposed two alternative approaches within the same framework. We also proposed metrics to evaluate the relevance and diversity of the expanded content which was shown to correlate well with human annotations.

Results show that automated expansion is indeed feasible and is a promising direction of research. Incorporating coherence of the expanded material appears to be the most promising future direction. We believe that automated text expansion will play a key role in smart authoring workflows for several domains in the near future.

References

1. Carbonell, J., Goldstein, J.: The use of MMR, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 335–336 (1998)
2. Fung, G., Rosales, R., Krishnapuram, B.: Learning rankings via convex hull separation. Advances in Neural Information Processing Systems, vol. 18 (2005)
3. Galgani, F., Compton, P., Hoffmann, A.: Combining different summarization techniques for legal text. In: Proceedings of the workshop on innovative hybrid approaches to the processing of textual data, pp. 115–123 (2012)
4. Li, Q., Candan, K. S., Qi, Y.: Extracting relevant snippets fromweb documents through language model based text segmentation. In: IEEE/WIC/ACM International Conference on Web Intelligence, pp. 287–290 (2007) doi: 10.1109/WI.2007.115
5. 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
6. Modani, N., Khabiri, E., Srinivasan, H., Caverlee, J.: Creating diverse product review summaries: A graph approach. In: International Conference on Web Information Systems Engineering, vol. 9418, pp. 169–184 (2015), doi: 10.1007/978-3-319-26190-4_12
7. Nenkova, A., McKeown, K.: Automatic summarization. Foundations and Trends in Information Retrieval, vol. 5, no. 2–3, pp. 103–123 (2011) doi: 10.1561/1500000015
8. Schlaefer, N., Chu-Carroll, J., Nyberg, E., Fan, J., Zadrozny, W., Ferrucci, D.: Statistical source expansion for question answering. In: Proceedings of the 20th ACM international conference on Information and knowledge management, pp. 345–354 (2011) doi: 10.1145/2063576.2063632
9. Taneva, B., Weikum, G.: Gem-based entity-knowledge maintenance. In: Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pp. 149–158 (2013) doi: 10.1145/2505515.2505715

ISSN 1870-4069

A Hybrid Approach to Extract Key Phrases from Arabic Text

Reda Ahmed-Zayed, Mohamed Farouk Abdel-Hady,
Hesham A. Hefny

Cairo University, Institute of Statistical Studies and Research,
Egypt

Reda_fcis@yahoo.com,
mohamed.abdel-hady@alumni.uni-ulm.de,
hehefny@ieee.org

Abstract: Key phrases are the phrases, consisting of one or more words, representing the important concepts in the document. This paper presents a hybrid approach to key phrase extraction from Arabic text. The proposed approach is an amalgamation of three methods: The first one is assigning weights to candidate key phrases based on term frequency and inverse document frequency, the second one is assigning weights to candidate key phrases using some knowledge about their similarities to the structure and characteristics of key phrases available in the memory (stored list of key phrases), and the third one is assigning weights to candidate key phrases using some knowledge about fatwa label (class or fatwa area). Also, an efficient candidate key phrase identification method has been introduced in this paper. The experimental results show that the proposed hybrid approach gives good performs.

Keywords: Arabic text mining, information retrieval, key phrase extraction automatic indexing, Arabic Islamic religion domain.

1 Introduction

The task of extracting key phrases from free text documents is becoming increasingly important as the uses for such technology expands. Key phrases are listed of phrases or key words composed of about five to fifteen important words and phrases that express the main topics discussed in a given document or article. Key phrases are useful for a variety of tasks such as text summarization, automatic indexing, clustering or classification, text mining.

It can provide the automation of generating metadata that gives a high-level description of a document's contents, highlighting important topics within the body of the text, summarizing documents for prospective readers, measuring the similarity between documents, making it possible to cluster and categorize documents, and searching more precise upon using them as the basis for search indexes or as a way of browsing a collection of documents [3].

Fig. 1. Key phrase extraction system overview.

When a Muslim has a question that they need to be answered from an Islamic point of view, they ask an Islamic scholar this question, and the answer is known as a" fatwa". It is similar to the issue of legal opinions from courts in common-law systems. A fatwa in the Islamic religion represents the legal opinion or interpretation that a qualified jurist or mufti can give on issues related to the Islamic law.

The fatwa request has to be directed to the most relevant mufti, and this task is first task we make in our proposed application [5]. Mufti start to answer the fatwa and manual assign key phrases for it. The main contribution of this paper is applying a hybrid approach for key phrase extraction using domain knowledge base and statistical information about the fatwa details depend on term frequency and inverse document frequency.

This paper and points to future work the most key phrase extraction systems which are proven to be successful have used supervised machine learning techniques. The main advantages of supervised machine learning techniques are that they can adapt to the specific nature of documents.

2 Related Work

Arora et al. [12] they proposed a new clustering algorithm based on the Kea Key phrase algorithm that used here to extract several Key phrases from source Text documents by using machine learning techniques.

The show that The Kea bisecting K-means clustering algorithm gives easy and efficient way to extract text documents from large amount of Text documents, there results showed that kea can an average match between one and two of the given key phrases chosen. The consistently good quality of the clustering that it produces, bisecting K-means is an excellent algorithm for clustering a large number of documents.

El-Beltagy et al. [2] presented the KP-Miner system, and demonstrated through experimentation and comparison with widely used systems that it is effective and efficient in extracting key phrases from both English and Arabic documents of varied length. Unlike other existing key phrase extraction systems, the KP-Miner system does not need to be trained on a particular document set in order to achieve its task. It also has the advantage of being configurable as the rules and heuristics adopted by the system are related to the general nature of documents and key phrases.

Gollapalli et al. [11] they explore a basic set of features commonly used in NLP tasks as well as predictions from various unsupervised methods to train their taggers. In addition to a more natural modeling for the key phrase extraction problem, they showed that tagging models yield significant performance benefits over existing state-of-the-art extraction methods.

3 System Overview

The architecture of the proposed key phrase extraction system is shown in Figure 1. The aim of this system is to automatically generate key phrase for a fatwa (legal opinion) requests. Each fatwa is associated with a category (Fatwa areas) by Muslim Scholar or our Proposed Routing System [5].

The Key phrase extraction in the proposed system is a two main phases. The first phase is text preprocessing and feature engineering for fatwa text to select the feature vector which represent each class this step is called Generating Knowledge Base and domain context. The second phase is a three-step process: candidate key phrase selection, candidate key phrase weight calculation and finally key phrase refinement. Each of these steps, is explained in more details about the employed algorithm.

4 Generating Knowledge Base and Domain Context

We need to build a bag of words for each fatwa class (category), to elect the list of words than can be considered as a feature vector for each category we need some process to extract this feature vector, these details explained in details in the next section.

4.1 Text Preprocessing

The nature of the Arabic text is different than the English text, preprocessing of the Arabic text is more challenging. A huge number of features or keywords in the documents lead to a poor performance in terms of both accuracy and time. Therefore, preprocessing is a very important step before training the text classifiers to get knowledge from massive data and reduce the computational complexity. Before Arabic word stemming step, fatwa requests are normalized as follows:

- Remove Fatwa Question introduction from start to word "المتضمن:".
- Remove punctuation.
- Remove special characters and remove any HTML tags.
- Remove diacritics (primarily weak vowels).
- Remove non-Arabic letters.
- Replace Arabic letter ALEF with hamza below, Arabic letter ALEF with madda above, and Arabic letter ALEF with hamza above with a Arabic letter ALEF.
- Replace final Arabic letter Farsi YEH with Arabic letter YEH.
- Replace final Arabic letter TEH marbuta with Arabic letter HEH.
- Stop-word removal: we determine the common words in the documents which are not specific or discriminatory to the different classes.
- Stemming: different forms of the same word are consolidated into a single word. For example, singular, plural and different tenses are consolidated into a single word.

Table 1. Number of fatwa belong to main fatwa class in the KP data set.

	Remove prefixes	**Remove Suffixes**
Light 1	ال،و ال،بال،كال،فال	None
Light 2	ال،و ال،بال،كال،فال،و	None
Light 3	،،	ه،ة
Light 8	،،	ها ، ان ، ات ، ون ، ين ، يه ، ية ، ه ، ة ، ي
Light 10	ال،و ال،بال،كال،فال،و ،مل،و	

Light Stemmer Larkey et al. [13] developed several light stemmers for Arabic, and assessed their effectiveness for information retrieval using standard TREC data. They have compared light stemming with several stemmers based on morphological analysis. The light stemmer, Light10, outperformed the other approaches. It has been included in the Lemur toolkit, and is becoming widely used for Arabic information retrieval. They tried several versions of light stemming, all of which followed the same steps:

– Remove Arabic letter WAW (and) for Light2, Light3, and Light8 if the remainder of the word is three or more characters long. Although it is important to remove Arabic letter WAW, it is also problematic, because many common Arabic words begin with this character, hence the stricter length criterion here than for the definite articles.
– Remove any of the definite articles if this leaves two or more characters.
– Go through the list of suffixes once in the (right to left) order indicated in figure below, removing any that are found at the end of the word, if this leaves 2 or more characters. The strings to be removed are listed in Fig. 2. The prefixes are actually definite articles and a conjunction.

4.2 Feature Engineering and Class Representation

Before any key phrase task or classification task, we need to represent each class (fatwa category) by feature vector. One of the most fundamental tasks that need to be accomplished is that of document representation and feature selection. We try to build lexicon for each class, each class can be reprinted by feature of vector. this vector is set of words; each text instance has to be represented as a fixed-length numeric feature vector which are mostly the text words.

This kind of text representation typically leads to high dimension input space. While feature selection is also desirable in other classification tasks, it is especially important in text classification due to the high dimensionality of text features and the existence of irrelevant (noisy or not important) features. Several methods are used to reduce the dimensionality of the feature space by choosing a subset of features in order to reduce the classification computational complexity without scarifying the accuracy. In this

paper, Chi-Squared ($\chi 2$) statistics [1] used as a scoring function to rank the features based on their relevance to the categories.

In general, text can be represented in two separate ways. The first is as a bag of words in which a document is represented as a set of words, together with their associated frequency in the document. Such a representation is essentially independent of the sequence of words in the document (context independent).

The second method is to represent each document as strings of words (called N-grams such as bigrams and trigrams), in which each document feature represents a sequence of words (it takes the context into consideration). In this paper, the bag-of-words representation is used as it has shown good key Phrase extraction performance.

5 Proposed Approach to Key Phrase Extraction

Proposed key phrase extraction consists of three primary components: document pre-processing we discuss at previous section, candidate key phrase identification and assigning scores to the candidates for ranking.

5.1 Candidate Key Phrase Identification

We follow a simple and knowledge poor approach to candidate key phrase identification is adopted as the first step of the proposed system. This approach is a variant of the candidate key phrase identification approach presented in [4]. A candidate key phrase is considered as a sequence of words containing no punctuations and stop words. A list of common verbs is also added to the stop word list because it is observed that the author assigned key phrases rarely contains common verbs. The process of candidate key phrase extraction has two steps:

Step1: extraction of candidate key phrases considering punctuations and stop words as the phrase boundary, Step2: Breaking further the phrases selected at the step one into smaller phrases using the following rules:

i. If a phrase is L -word long, all n-grams (n varies from 1 to L-1) are generated and added to the candidate phrase list,

ii. If a phrase is longer than five words, it is discarded.

Figure 3 shows a sample sentence and the candidate Key phrases identified from this sentence. Some candidate phrases generated using the above-mentioned method may not be meaningful to human readers. For example, in figure1, the candidate phrase "بصفة دائمة" is less meaningful. After computing phrase frequency and phrase weight, such kind of candidate key phrases are filtered out. For this purpose, some conditions are applied.

Condition one is to choose threshold on the phrase weight (Phrase weighting scheme has been presented in the next subsection which is a function of phrase frequency, inverse document frequency, domain knowledge etc. The second condition is related to the first appearance of the phrase in the document. Previous works [9] have suggested that key phrases appear sooner in an article. The works in El-Beltagy [2] states that a phrase occurring the first time after a predefined threshold is less likely a key phrase.

Sample Fatwa

بسم الله الرحمن الرحيم. الحمد لله وحده والصلاه والسلام علي من لا نبي بعده سيدنا محمد رسول الله وعلي اله وصحبه ومن تبعه باحسان الي يوم الدين. اطلعنا علي الطلب المقدم من/ محمود عزمي احمد ابو العزم المقيد برقم 126 لسنه 2006 م المتضمن: أُجريت لي عمليه جراحيه في البروستاتا والمثانه مما ادي بعد الشفاء من الجراحه الي خروج قطرات بول مني بصفه دائمه وعدم التحكم فيه بعد الاستنجاء، مما يضع النفس في حيره وشك في الوضوء والصلاه . نرجو الافاده ، وكيف يصح الوضوء والصلاه ؟

Fatwa Question without introduction

أُجريت لي عمليه جراحيه في البروستاتا والمثانه مما ادي بعد الشفاء من الجراحه الي خروج قطرات بول مني بصفه دائمه وعدم التحكم فيه بعد الاستنجاء، مما يضع النفس في حيره وشك في الوضوء والصلاه . نرجو الافاده ، وكيف يصح الوضوء والصلاه ؟

Initial list of candidate key phrases (after step1).

اجريت,عملية جراحيه ,البروستاتا والمثانه ,ادي ,الشفاء ,الجراحه,خروج قطرات بول ,بصفة دائمة , التحكم ,الاستنجاء,يضع النفس,حيرة وشك ,الوضوء.

The list of candidate phrases (after step2).

اجريت,عمليه,جراحيه ,عمليه جراحيه,البروستاتا,والمثانه,البروستاتا والمثانه,ادي,الشفاء,الجراحة,خروج,قطرات, بول ,خروج قطرات,خروج قطرات بول,قطرات بول,بصفه,دائمه,بصفه دائمه,التحكم,الاستنجاء,يضع,النفس,يضع النفس,حيره,وشك,حيره وشك,الوضوء

Fig. 2. A sample fatwa and the candidate key phrases identified from this fatwa.

A threshold is set on the position of the phrases where the phrases are numbered sequentially and the first phrase in the document is numbered as 1 and the last phrase is numbered as N. If a phrase appears first after the given threshold it is ignored, that is, if a phrase X appears first at pos i and the threshold value is set to T_{pos} and $i > T_{pos}$, the phrase X is discarded.

5.1.2 Assigning Scores to Candidate Key Phrases

In general, a fatwa has a few numbers of author assigned key phrases. To select a small subset of candidates as the key phrases requires assigning weights to the candidates and raking them based on these weights.

5.1.2.1 Assigned Score Phrase Frequency, Inverse Document Frequency

The weight of a candidate key phrase is computed using three important features: phrase frequency, inverse document frequency and domain specificity. Weighting using phrase frequency (PF) and inverse document frequency (IDF). The score for a candidate key phrase due to PF and IDF features is computed using the following formula:

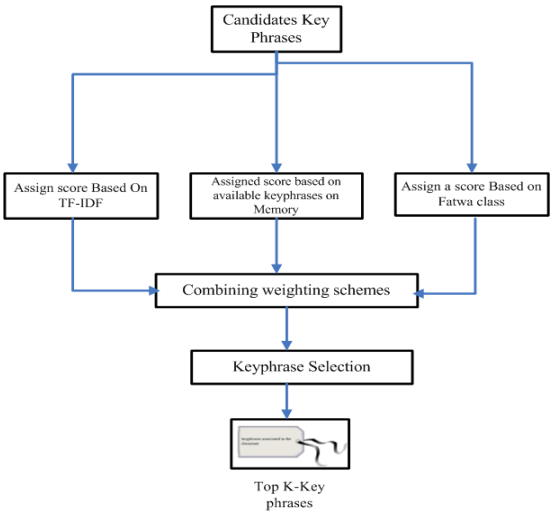


Fig. 3. Assign score to candidate key phrases.

$$Score PF \times IDF = \begin{cases} PF \times IDF, if\ plength = 1, \\ PF \times log(N), if\ plength > 1, \end{cases}$$

where:

$plength$= length of the phrases in terms of words PF = phrase frequency which is counted as number of times a phrase occurs in a document.

IDF= log(N/DF), where N is the total number of documents in the corpus (a collection of documents in a domain under consideration) and DF is the number of documents in which a phrase occurs at least once. Equation (1) shows that for multi-word phrases, phrase score is computed using PF × log (N), which is basically PF * IDF with DF set to 1. This is due to the fact that multi-word phrases do not occur as frequently within a relatively small collection of documents as do single-word phrases.

5.1.2.2 Using Domain Knowledge for Weighting Vandidate Key Phrases

We assigned score to each candidate key phrase using two condition, the first condition depend on if the key phrase word appear in the list of words that represent the class of fatwa as we mention the previous section, and the second condition depend on if the key phrase word appear in the list of key Phrase in the memory. This list contains manual key phrase extracted from fatwa and all possible key phrases generated from fatwa title.at next section we will discuss how assign weight to candidate key phrase in details.

5.1.2.3 Assigned Score based on Available Key Phrases on Memory

A score is assigned to a candidate key phrase based on how much it is similar to the structure and characteristics of key phrases available in the memory (a stored list of key phrases for the domain under consideration). For this purpose, a key phrase list is created with readily available author assigned key phrases collected from predefined key phrase for Fatwa. A predefined key phrase list is used to create a domain specific

glossary database giving some knowledge about the structure and characteristics of key phrases.

A list of key phrases is collected for creating glossary database are not included in the set of fatwas (the test set) on which the proposed key phrase extraction system is tested. However, using such a list of key phrases stored in the memory for weighing the candidate key phrases can be considered as some sort of partial supervision provided to the key phrase extraction system. The use of this kind of knowledge base in key phrase extraction task has previously been investigated in Sarkar [6, 10]. A variant of the method presented in Wu [10] is used for the proposed domain specific key phrase extraction task.

From the key phrase list, two tables are created: table1 is the keyword table, which is created by splitting the key phrases belonging to the key phrase list into words that can be called as keywords. This table has two columns (keyword, weights) and table2 is key sub-phrase table which consists of all sub phrases generated from the key phrases in the key phrase list. For any manual key phrase in the key phrase list, all possible n-grams (n varies from 2 to n) are generated and included in key sub-phrase table. The key sub-phrase table has also two columns (sub-phrase, weights) [6].

Weights for keywords in the keyword table are assigned using the following rules:

- If a keyword appears always alone independently in the key phrase list it is assigned a score of 1.
- If the keyword appears always as part of another key phrase, that is, if it has no independent existence in the key phrase list, it is assigned a score, which is computed as 1/log(c), where c is the number of times the keyword appears as the part of key phrases. Here it is assumed that a keyword, which has no independent existence and repeats many times in the key phrase-list only as the parts of other key phrases, is less domain specific.
- If the keyword appears independently in the key phrase list in some cases and also appears as part of key phrases in some other cases, it is assigned a score which is computed based on the formula: 0.5 × (1 + 1 / log (c)), where c is the number of times the keyword occurs as the part of key phrases.

The Weight of a sub-phrase or a phrase in the key sub-phrase table is computed by summing up the weights of the keywords of the sub-phrase or the phrase. The keyword table and the key sub-phrase table are used as domain knowledge in computing a score for a candidate key phrase. The score for a candidate phrase is computed using the following equation:

$$score_{D} = \sum_{i=1}^{m} K_i + \sum_{j=1}^{pc}, \tag{1}$$

where:

K_i = the weight of the $i-th$ keyword in the candidate keyphrase.
P_j= weight of the $j-th$ sub-phrase associated with candidate keyphrase.
M = the number of keywords in a candidate keyphrase.
PC= the number of sub-phrases generated from the candidate keyphrase.

The sub-phrases of the candidate key phrase are generated by computing all possible n-grams, where n varies from two to length of the phrase. When n is set to the length of the candidate key phrase, the n-gram is the candidate key phrase. The reason for taking the weights of all possible sub-phrases in calculating the candidate key phrase score, in addition to the weights of individual words, is to decide whether a sub-phrase is a manual key phrase in the key phrase table.

If it is, this candidate key phrase is assumed more important. This feature will favor those candidate key phrases which itself or whose parts are found in the knowledge base. Availability of a phrase or its sub-phrases in the knowledge base provides some evidence in support of key phrase worthiness of a phrase. Thus, with this knowledge base, the proposed key phrase extraction system is provided with some sort of partial supervision.

5.1.2.4 Assign a Score based on Fatwa Class

A score is assigned to a candidate key phrase based on fatwa class. For this purpose, a list of words is selected as feature vector which represent each class. Every class repented by 100 words, we make a feature selection using TF×IDF method. Each word repents presenting the class, for example word"المياة " represent class "الطهارة" by 100% and "الصلاة" by 80% and so on. Weights for keywords in the keyword table are assigned using the following rules:

– For each word in a candidate key phrase if the word appears in fatwa the weight formula ass following:

$$Score_C = \sum_{i=1}^{m} \frac{w[i]_{idf(c)}}{max\ w_{idf(c)}}, \tag{2}$$

where w_{tf} : the idf of word of i in the given class.

max w_{tf} : The max idf value for words in the given class words

5.1.2.5 Combining Weighting Schemes

The three weighting schemes have already been discussed in the previous subsections. These three types of scores should be combined to assign a unique score to each candidate key phrase. The combined score for a candidate key phrase is computed using the following linear combination of three scores:

$$SCORE = \alpha \times Score_{pf \times idf} + (1 - \alpha) Score_D\ \alpha \times Score_K +, \tag{3}$$

where:

$Score_{pf*idf}$ The score based on phrase frequency and inverse document frequency computed using the equation (1).

$Score_D$ The score based on domain specificity of a phrase computed using the equation (2).

$Score_c$ The score based on domain specificity of a phrase computed using the equation (3).

α Is the tuning parameter whose value is decided through experimentation the best results are obtained when α is set to 0.6 [6].

5.1.2.6 Extracting and Select Key Phrases

After assigning scores to the candidate key phrases, the next step is to select K top-ranked candidate key phrases as the final list of key phrases. The user can specify the value of K.

5.1.2.7 Performance Evaluation

The dataset used in the experiments was provided by the Egyptian Dar al-Ifta. Dar al-Ifta al Misryyah[1] started as one of the divisions of the Egyptian Ministry of Justice. In view of its consultancy role, capital punishment sentences among others are referred to the Dar al-Ifta al- Misryyah seeking the opinion of the Grand Mufti concerning these punishments. The dataset contains about 100,000 text instances

In order to further reduce the computational complexity of classification; feature selection was applied as follows: We used $Chi - square$ feature-ranking method. Separately for each main label (fatwa class) in order to obtain a ranking of all features for that label.

We then selected the top 100 features for each label. After the aforementioned preprocessing, and the removal of empty examples (examples with no features or labels, or multi question fatwa) the final version of the dataset included 1215 instances. The following table represents each main label and related fatwa. Each label has numbers of child we ignore the child labels and combine fatwa to the parent node of the leaf.

We merge the leaf nodes that does not contain any child in the parent node and consider the parent is the fatwa label. When we try to extract the key phrase, we have the fatwa label from the fatwa routing system. according this label, we get the label words.

Each word has a value that value represents a certain percentage in a certain class. This value we make consideration when we calculate the word weight in the candidate key phrase.

6 Experiments and Results

For extracting the key phrases from a test fatwa, the proposed system identifies first the candidate key phrases, computes phrase weight using the equation (3), filters out noisy phrases based on two conditions discussed in the previous section and assigns scores to the remaining candidate key phrases. Finally, the top-ranked K candidate key phrases are selected as key phrases.

To filter out noisy phrases, two conditions discussed in subsection 3.2 are applied here in the pre-specified order as follows: (1) Phrases whose position of the first occurrence in the document is greater than $Tpos$ are discarded. The value of Tpos is set to 100 to obtain the best results on the dataset used for the proposed work, (2) the threshold value on the phrase weight is adjusted to keep those candidate key phrases which occurs at least twice in a document or which has higher similarity to the phrases in the manually created knowledge base.

[1] http://dar-alifta.org

Table 2. Number of fatwas belong to main fatwa class in the KP data set.

Main Class Name	Arabic Name	Fatwa count
Purity(alttahara)	الطهارة	77
Prayer(alssala)	الصلاة	224
Funerals(aljanayiz)	الجنائز	123
Zakat(alzzaka)	الزكاة	537
Fasting(alssiam)	الصيام	74
Hajj and Umrah(alhajj waleumra	الحج والعمرة	104
Dikher and pray(aldhdhikr walddiea)	الذكر والدعاء	146

Result of Top 5 key Phrase.

Class	Purity	Prayer	Funeral	Zakat	Fasting	Hajj	Dikher & pray
Precision	90.15%	91.35%	85.32%	90.15%	74.1%	82.41%	60.23%
Recall	89.34%	87.21%	82.53%	92.65%	86.85%	83.73%	67.62%

Result of Top 10 key Phrase.

Class	Purity	Prayer	Funeral	Zakat	Fasting	Hajj	Dikher & pray
Precision	95.2%	92.5%	90.12%	96.25%	78.36%	85.32%	75.15%
Recall	96.5%	90.25%	88.23%	95.1%	80%	86.23%	70.32%

Result of Top 15 key Phrase.

Class	Purity	Prayer	Funeral	Zakat	Fasting	Hajj	Dikher & pray
Precision	97.35%	96.1%	93.62%	96.95%	84.56%	92.45%	86.15%
Recall	97.3%	93.1%	91.33%	96.63%	86.72%	89.11%	76.42%

We compare the output result to the mufti assigned key phrase the similarity between automated key phrase extraction and manual key Phrase and the flowing tables show the result of Performance over combined keywords when extracting, 5, 10, and 15 key phrases.

7 Conclusion

This paper discusses a hybrid key phrase extraction approach in the Islamic fatwa domain. The proposed approach combines domain knowledge with the features namely phrase frequency, inverse document frequency and phrase position in a more effective way. The proposed approach results in an easy-to-implement key phrase extraction system that outperforms some state-of-the art key phrase extraction systems. The experimental results also suggest that the proposed key phrase extraction method is effective in Islamic fatwa domain and incorporation of domain knowledge as partial supervision boosts up the system performance.

References

1. Mesleh, A.: Chi square feature extraction based SVMs arabic language text categorization system. Journal of Computer Science, vol. 3, no. 6, pp. 430–435 (2007)
2. El-Beltagy, S. R., Rafea, A.: KP-miner: A keyphrase extraction system for english and arabic documents. Information Systems, vol. 34, no. 1, pp. 132–144 (2009) doi: 10.1016/j.is.2008.05.002
3. El-Shishtawy, T., Al-Sammak, A.: Arabic keyphrase extraction using linguistic knowledge and machine learning techniques. arXiv preprint arXiv, 1203.4605 (2012) doi: 10.48550/arXiv.1203.4605
4. Kumar, N., Srinathan, K.: Automatic keyphrase extraction from scientific documents using N-gram filtration technique. In: Proceedings of the eighth ACM symposium on Document engineering, 199–208 (2008) doi: 10.1145/1410140.141018
5. Zayed, R. A., Hady, M. F. A., Hefny, H.: Islamic fatwa request routing via hierarchical multi-label arabic text categorization. In: Proceedings of First International Conference on Arabic Computational Linguistics (ACLing), IEEE pp. 145–151 (2015) doi: 10.1109/ACLing.2015.28
6. Sarkar, K.: A hybrid approach to extract keyphrases from medical documents. arXiv preprint arXiv, pp. 1303–1441 (2013) doi: 10.5120/10565-552
7. Turney, P. D.: Extraction of keyphrases from text: evaluation of four algorithms (1997) doi: 10.48550/arXiv.cs/021201
8. Turney, P. D.: Learning algorithms for keyphrase extraction. Information retrieval, vol. 2, no. 4, pp. 303–336 (2000) doi: 10.1023/A:1009976227802
9. Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., Nevill-Manning, C. G.: KEA: Practical automatic keyphrase extraction. In: Proceedings of the fourth ACM conference on Digital libraries (1999)
10. Wu, Y. F. B., Li, Q.: Document keyphrases as subject metadata: incorporating document key concepts in search results. Information Retrieval, vol.11, no.3, pp. 229–249 (2008) doi: 10.1007/s10791-008-9044-1
11. Das-Gollapalli, S., Li, X. L.: Keyphrase extraction using sequential labeling. arXiv preprint arXiv:1608.00329 (2016) doi: 10.48550/arXiv.1608.00329
12. Arora, A., Er-Abhishek, C.: Keyphrase extraction algorithm, PARIPEX-Indian Journal of Research, vol. 5, no. 4 (2016)
13. Larkey, L. S., Ballesteros, L., Connell, M. E.: Improving stemming for Arabic information retrieval: Light stemming and co-occurrence analysis. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Rerieval (SIGIR 2005), pp. 275–282 (2005) doi: 10.1145/564376.5644

ISSN 1870-4069

Hidden Recursive Neural Network for Sentence Classification

Minglei Li[1], Qin Lu[1],
Yunfei Long[1], Lin Gui[2]

[1] The Hong Kong Polytechnic University,
Department of Computing,
Hong Kong

[2] Harbin Institute of Technology,
Laboratory of Network Oriented Intelligent Computation,
China

{csmli, csluqin, csylong}@comp.polyu.edu.hk,
guilin.nlp@gmail.com

Abstract. Recursive Neural Network has been successfully used in sentence-level sentiment analysis for language compositionality based on structured parsing trees. Later on, several modified versions are proposed. These models either treat word vectors as model parameters or employ pre-trained word vectors as input. The former has the advantage of learning task specific word vectors but has much larger parameter size. The later has the advantage of using the encoded semantic information in the vectors and has much smaller parameter size but the general word vectors may be not task-specific. In this work, we propose a hidden recursive neural network (HRNN) which can take the advantages of both learning word vectors and using pre-trained word vectors. This model takes the pre-trained word vectors as the input and adds one hidden layer to extract task-specific representation. Then the recursive composition process is performed in the hidden space. We perform extensive experiments on several sentence classification tasks and results show that our proposed model outperforms both methods and the other baselines, which indicates the effectiveness of our proposed model.

Keywords: Recursive neural network, word vectors, recurrent neural networks.

1 Introduction

Sentence classification requires appropriate feature representations. Traditional methods are mainly based on manually defined features such as bag-of-words, sentiment lexicon, n-grams, etc [17,31]. In recent years, word vector representation, obtained through neural network as a dense and low dimensional vector in an unsupervised way, shows promising result on many tasks of natural language processing [3,14]. The word vectors can encode semantic information.

Inspired by the word vector representation, different methods are proposed to infer dense vector representation of longer text, such as phrases and sentences, and then apply it to task specific sentence-level classification problems such as the sentence-level sentiment analysis. Recursive neural network (RNN) shows promising result on inferring semantic representation of longer text units based on structured parsing trees [26].

This model computes phrase and sentence representation recursively in a bottom-up approach based on syntactic parsing trees. Several modified versions are proposed based on the original RNN, such as the Matrix-vector Recursive Neural Network [25], the Recursive Neural Tensor Network [28], the Adaptive Recursive Neural Network (AdaRNN) [5], the deep recursive neural network (DRNN) [10], the tagging-specific recursive neural network [23], and the Gated Recursive Neural Network (GRNN) [2], etc.

Some models, such as the RNN, MV-RNN, RNTN and AdaRNN, treat the word vectors as model parameters and the word vectors are learned during training the model so that the learned word vectors are more task specific. However, this will make the model parameters increase linearly with the vocabulary size, which will need much larger training data size. That is why the vector dimension is set to about 25 in the original RNN, MV-RNN and RNTN, much smaller compared to the dimension of the commonly used pre-trained word vectors.

In contrast, some models, such as the DRNN and GRNN, use pre-trained word vectors as input, which can reduce the model parameters and this also helps to make full use of the encoded semantic information in the word vectors. However, pre-trained word vectors are designed for general semantic representation, which cannot contain sufficient task specific knowledge compared to learning the word vectors as model parameters.

Intuitively, if the semantic meaning of a word is fully encoded into a dense vector, we should be able to infer the task specific subspace representation, and then perform the recursive composition in this subspace. The hierarchical deep learning model has been shown to be able to extract higher level abstract representation [13]. Based on the above analysis, we propose a new RNN model to take the advantages of both learning the word vectors and using pre-trained word vectors.

This model employs the pre-trained word vectors as input to make use of the semantic information encoded in the vectors and adds one hidden layer beyond the input layer to extract task specific information. Then, the recursive composition is performed in the hidden space. We perform extensive experiments on several sentence classification tasks and the results indicate that our proposed model outperforms both the RNN model that learns the word vectors and the RNN model that simply employs pre-trained word vectors as input. Our model also outperforms other baselines on most of the datasets.

The rest of the paper is organized as follows. Section 2 introduces related work on sentence level classification problem, especially for the sentence representation learning. Section 3 goes into the details of the original RNN model, and based on this, we introduce the details of our model in section 4. Section 5 shows the experiments and result analysis. Section 6 gives the conclusion and future work.

2 Related Work

All classification models at the sentence level are based on sentence representation. Traditional representation methods are mainly based on manual features, such as n-grams, word lexicon, POS tags or manual defined rules [24,17]. These methods treat a word as a symbol and cannot consider the relationship between words, such as antonyms and synonyms. They also fail to consider the word order information. The word vector (also called word embedding) can encode the semantic information of a word into a low-dimensional and dense vector, such as it can encode the following relationships between the corresponding word vectors: *"king"-"queen"="man"-"woman"* or *"China"-"Beijing"="France"-"Paris"* [14]. Such word representation can be used to learn higher level representations such as a phrase or a sentence.

Previous methods to infer phrase level representation from word representation include vector average, addition, and element-wise multiplication [15,16]. Baroni also considers POS categories of words that a noun is used as a vector whereas adjectives, adverbs, and verbs form a matrix to modify the properties of the noun [1]. Recently, neural network based models are proposed. For example, the recurrent neural network is used to infer sentence representations for sentiment classification [11].

The convolutional neural network is employed for sentence classification [12]. This model does not consider the syntactic structure information of a sentence. Based on the structured parsing tree, the recursive neural network (RNN for short) model represents a tree node as a vector and the parent node is computed from the child nodes through a matrix composition function [26]. The Matrix-Vector Recursive Neural Network (MV-RNN) modifies the RNN by representing a node through a vector and a matrix so that it can consider the relation between the child nodes. In Recursive Neural Tensor Network (RNTN), the composition function is a global tensor instead of a matrix [28].

The Adaptive Recursive Neural Network (AdaRNN) employs n composition function during the recursive process and the parent node's representation is the weighted addition of the n composition functions [5]. This model treats the word vectors as parameters without employing the pre-trained word vectors. Distinguishing the phrase types, the Phrase Recursive Neural Network (PRNN) uses two sets of composition functions for the inter-phrases and outer-phrases respectively [19]. This model takes pre-trained word vectors as input without learning the word vectors.

Taking the syntactic information into consideration, Qian [23] proposes the POS tagging-specific recursive neural network that learns different composition functions for different kinds of POS tagging and also learns tagging-specific embedding to be concatenated after the word embedding during the composition process. This model also employs pre-trained word vectors as input. A gated recursive neural network is proposed that employs a full binary tree structure to control the composition process through two kinds of gates [2].

3 The Recursive Neural Network Model

The original RNN is used for sentence-level sentiment analysis task, however, it can be used in any sentence-level classification task.

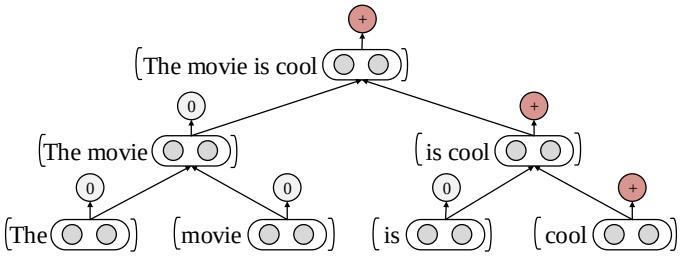


Fig. 1. RNN for sentiment analysis. It is trained based on the constraint of the sentiment label of every node. The word vectors are treated as model parameters and are learned simultaneously during training the model.

For easy discussion, we will take the sentiment analysis as a classification example to explain the model. The structure of RNN [26] for sentiment analysis is shown in Figure 1. Given a structured representation of a sentence, such as the syntactic parsing tree, RNN computes the node's representation in a bottom-up style. To be more specific, given the vector representations of the left child $\boldsymbol{v}_l$ and the right child $\boldsymbol{v}_r$, the representation of its parent $\boldsymbol{v}_p$, is computed using the following formula:

$$\boldsymbol{v}_p = \sigma\left(W\begin{bmatrix}\boldsymbol{v}_l\\ \boldsymbol{v}_r\end{bmatrix} + \boldsymbol{b}\right), \tag{1}$$

where $W \in \mathbb{R}^{d\times 2d}$ and $b \in \mathbb{R}^{d\times 1}$ are the composition parameters shared by all the nodes; $\boldsymbol{v}_l$ and $\boldsymbol{v}_r \in \mathbb{R}^{d\times 1}$; d is the dimension of the nodes' vectors; σ is the activation function. In the original RNN, the vector of the leaf node is treated as model parameters and is also learned during training the model. So the leaf nodes' vectors consist of a parameter matrix $L \in \mathbb{R}^{|V|\times d}$ where $|V|$ is the vocabulary size of the training data. After obtaining the node's vector representation, the node's label is predicted through softmax function:

$$\boldsymbol{y}_i = g\left(W_s\boldsymbol{v}_i + \boldsymbol{b}_s\right), \tag{2}$$

where $\boldsymbol{y}_i \in \mathbb{R}^K$ is the output class vector and K is the class number; g is the softmax function; $W_s \in \mathbb{R}^{K\times d}$ and $\boldsymbol{b}_s \in \mathbb{R}^{K\times 1}$ are the weight matrix and bias, respectively. The original model predicts every internal node's label to constrain the recursive process. The loss function consists of cross entropy and L2 regularization:

$$J(\theta) = -\sum_{i}^{m}\sum_{j}^{K} t_j log(y_j) + \lambda\left\|\theta\right\|^2, \tag{3}$$

where t_j and y_j are the gold label and predicted label respectively; K is the class number; m is the training sample number; $\theta = \{W, W_s, b, b_s, L\}$ is the set of model parameters; λ is the regularization parameter.

The original RNN model treat the word vectors as model parameters, which can learn more task-specific word vectors.

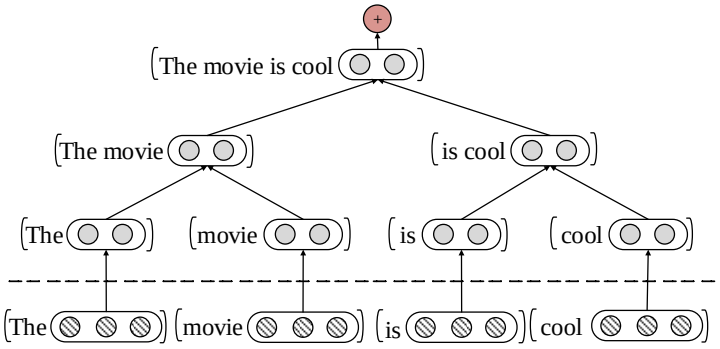


Fig. 2. HRNN for sentiment analysis. Pre-trained word vectors (below the dotted line) serves as the input and one hidden layer is added(above the dotted line) as the representation of the leaf nodes.

However, this will make the parameters space increase linearly with the vocabulary size. The pre-trained word vectors, such as the public available word2vec [14] or Glove [22], have also been used as input vector of the leaf nodes in the latter work [10,29,2]. This can make use of the encoded semantic information in the word vectors but the pre-trained word vectors are designed for the general word meaning, which can not be task-specific.

4 Our Proposed Model

To be able to employ the advantage of learning task-specific representation and the pre-train word vectors, we propose our model as shown in Figure 2. Compared to RNN, we employ the pre-trained word vectors as input and add one hidden layer to extract the task-specific representation, which can not only make use of the semantic information encoded in the word vectors, but also learn the task-specific representation without increasing the model parameters.

For this reason, we refer to our proposed model as the Hidden Recursive Neural Network (HRNN). In the HRNN model, the representation of a leaf node j is calculated by:

$$\boldsymbol{v}_j = \sigma\left(W_I \boldsymbol{v}_j^I + \boldsymbol{b}_I\right), \tag{4}$$

where $\boldsymbol{v}_j^I$ is the pre-trained word vector of word j, W_I and $\boldsymbol{b}_I$ are the model parameters shared between the input layer and the hidden layer to extract task-specific information, and σ is the activation function. The representations of all non-leaf nodes are calculated using Formula 1.

In addition, during the training process, the recursive process of the original RNN is constrained by the sentiment label of the internal nodes because it predicts every node's label. This limits the model's applications because it can only be trained on the corpus of the Stanford Sentiment Treebank (SST) [28], which is the only fully annotated corpus at every phrase.

This makes it difficult to be applied to other sentence classification tasks as annotating every node in the trees is very labor intensive. To extend this model to other tasks, we remove the label constraint of the internal nodes and only predict the label of the root node.

Intuitively, if the pre-trained word vectors have encoded the general semantic meaning and the recursive process can infer the higher phrases' representation, the constraint of task-specific label (such as sentiment label) can make the higher phrases' representation deviated from the original semantic meaning and this may have adverse effect on the final sentence representation because the deviation may accumulate during the recursive process.

If this intuition is true, annotating all the phrases in the parsing tree is not worthy. We will perform experiments to test the effect of this simple modification. The output of the root node is calculated using softmax:

$$\boldsymbol{y} = g(W_s \boldsymbol{v}_{\text{root}} + \boldsymbol{b}_s), \tag{5}$$

where $\boldsymbol{v}_{\text{root}}$ is the root node's vector representation and g is the softmax function. The loss function is the same as Formula 3 and $\theta = \{W_I, W, W_s, b_I, b, b_s\}$.

5 Performance Evaluation

We conduct two experiments to test the effect of our two modifications. The first experiment is designed to investigate the effect of removing the label constraint of the internal nodes. The second experiment is designed to test the effectiveness of our proposed model that employs the pre-trained word vectors and adds one hidden layer.

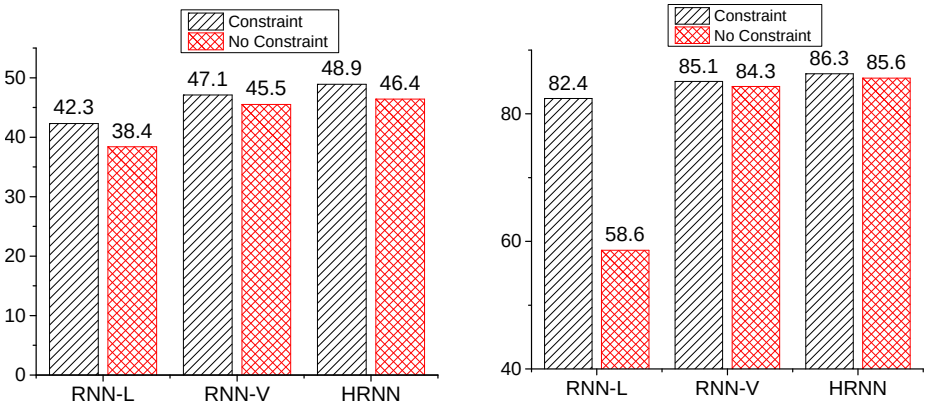
Parameter Setting. For the activation function, we use the rectifier linear activation $f(x) = max\{0, x\}$, which shows better result than Sigmoid function in previous work [10]. For model training, we use back propagation [7] through the stochastic gradient descent algorithm of mini-batch AdaGrad [6].

The original learning rate for the hyper-parameter is set to 0.01 and the regularization parameter λ is set to 0.0001. The mini-batch size is set to 50 and maximum epoch number is set to 100. For the word vectors, we test a number of pre-trained word vectors such as the 300-dimensional Google word vector[1] trained on Google News dataset [14], and Glove with different dimensions trained on different datasets [22].

The 300-dimensional Glove840B trained on 840 billion tokens of common crawl data[2] achieves the best result, so we choose this in the following experiments. For unknown words, we randomly initialize them. No fine-tuning is performed on the pre-trained vectors like the work in [12] because we assume that the pre-trained word vectors represent the general word meanings and the hidden layer is responsible for sentiment specific information extraction.

[1] code.google.com/p/word2vec/

[2] nlp.stanford.edu/projects/glove/



(a) Result for the fine-grained task.

(b) Result for the binary task.

Fig. 3. The performance (accuracy) of RNN under different settings with or without the label constraint. **RNN-L**: The RNN model that learns the word vectors. **RNN-V**: The RNN model that uses the pre-trained word vectors as input. **HRNN**: Our proposed model. The black box (Constraint) is the result with label constraint and the red box (No Constraint) is the result without label constraint.

5.1 Experiment 1

This experiment aims to test the effect of removing the label constraint of the internal nodes. We test the performance of the RNN-based models with or without the label constraint.

The tested models include the RNN model that treats word vectors as model parameters (denoted as RNN-L, where L means learning the word vectors), the RNN model with the pre-trained word vectors as input (denoted as RNN-V where V means employing pre-trained vectors as input), and the proposed HRNN model that employs pre-trained word vectors as input and adds one hidden layer.

The word vector dimension of RNN-L is set to 25. The hidden dimension of HRNN is set to 100.

Since only the Stanford Sentiment Treebank (SST) [28][3] is annotated at every node, we perform the experiment on this dataset. The SST comes from critic reviews in Rotten Tomatoes and every sentence is parsed by the Stanford Parser[4].

Then, every phrase in the parsing trees is annotated with polarity through the crowdsourcing platform of Amazon Mechanical Turk. There are two tasks for this dataset. The first is binary sentiment classification on positive/negative. The second is more fine-grained classification on five classes: very negative, negative, neutral, positive, and very positive. We use the standard train/dev/test splits of 6920/872/1821 for the binary classification and 8544/4404/2210 for the fine-grained classification. The result is shown in Figure 3. As shown in the figure, for different models, the result with the label constraint all outperforms the result without the label constraint, which

[3]http://nlp.stanford.edu/sentiment/treebank.html

[4]http://nlp.stanford.edu/software/lex-parser.shtml

Table 1. The parameter size of different models. d is the dimension of node's vector. For RNN-L, $d = 25$, for RNN-V, $d = 300$, for HRNN, $d = 100$. $d_v = 300$ is the dimension of pre-trained word vectors. $|V| = 18K$ is the vocabulary size (We use the vocabulary of the fine-grained task as an example).

Model	Model Size	#of parameters
RNN-L	$\|V\| \times d + 2d \times d$	451K
RNN-V	$2d \times d$	180K
HRNN	$d_v \times d + 2d \times d$	50K

indicates that the label constraint of the internal nodes is beneficial for training the RNN model. However, obtaining such training data is labor intensive. Comparing between the RNN-L, RNN-V and HRNN shows that learning the vectors achieves the worst result, this may result from the much larger parameter size.

RNN with pre-trained word vectors (RNN-V) performs better because of the pre-trained word vectors. HRNN further outperforms RNN-V, which indicates the effectiveness of the added hidden layer. We will conduct more experiments to validate this. From this experiment, we can conclude that the label constraint is beneficial for the RNN model and our model performs better than the original RNN model.

The detail of the different models' parameters is shown in Table 1. In this table, the *Model Size* is the model parameter numbers. Since the softmax weights and bias of different models are the same, we just ignore this part and only calculate the composition and word vector part. It shows that the HRNN model has much fewer parameters than the RNN-L and RNN-V.

5.2 Experiment 2

The second experiment aims to test the effectiveness of our proposed model on other sentence level classification problems. The evaluation datasets include:

1. MPQA [32]:Binary classification of short texts with positive and negative[5].
2. Subj [20]: Classifying sentence as subjective or objective[6].
3. MR [21]:Binary classification of movie reviews with positive and negative[7].
4. CR [9]: Binary classification of customer review with positive and negative[8].

The detailed statistic information of these datasets is shown in Table 2. Since all these datasets only have the labels at the sentence level, all the RNN-based models are trained without the label constraint of the internal nodes. To be noted that the best hidden dimension d varies for different datasets and we only report the best result under the best hidden dimension. The best d is 100 for MPQA, 90 for MR, 100 for Subj and 90 for CR. HRNN is compared with the following baselines:

[5]Opinion polarity detection subtask of the MPQA dataset
[6]www.cs.cornell.edu/people/pabo/movie-review-data/
[7]www.cs.cornell.edu/people/pabo/movie-review-data/
[8]www.cs.uic.edu/ liub/FBS/sentiment-analysis.html

Table 2. Statistics of the datasets. K is the class number. l is the sentence average length of the dataset. N is the total sample number. $|V|$ is the vocabulary size. $|V_{pre}|$ is the number of words present in the pre-trained word vectors. $Test$ is the training the test data size (CV means 10-fold cross validation).

Dataset	K	l	N	$\|V\|$	$\|V_pre\|$	$Test$
MPQA	2	3	10606	6225	6205	CV
Subj	2	23	10000	22361	20898	CV
MR	2	20	10662	19994	18632	CV
CR	2	19	3771	5624	5481	CV

1. **RNN-V**: The original RNN model with pre-trained word vectors as input and without hidden layer.
2. **RNN-L**: The original RNN model that treats the word vectors as model parameters.
3. **CNN-static** [12]: Using convolutional neural network that employs pre-trained word vectors as input and the word vectors are fixed without fine-tuning.
4. **CNN-non-static** [12]: Using convolutional neural network that employs pre-trained word vectors as input and the word vectors are fine-tuned during training the model.
5. **CNN-multichannel** [12]: Same architecture as CNN-non-static but with two sets of pre-trained word vectors.
6. **RAE** [27]: Recursive autoencoders with pre-trained word vectors trained from Wikipedia.
7. **CCAE** [8]: Combining the power of recursive, vector-based models with the linguistic intuition of the CCG formalism.
8. **Sent-Parser** [4]: Sentiment analysis specific parser that directly analyzes the sentiment structure of a sentence.
9. **NBSVM** and **MNB** [31]: Naive Bayes SVM and Multinomial Naive Bayes with the variant of uni-bigrams.
10. **G-Dropout** and **F-Dropout** [30]: Gaussian dropout and fast dropout.
11. **Tree-CRF** [18]: A dependency tree based method for sentiment classification of subjective sentences using conditional random fields with hidden variables.
12. **CRF-PR** [33]: Encoding the intuitive lexical and discourse knowledge as expressive constraints and integrating them into the learning of the conditional random filed model via posterior regularization.

The results are shown in Table 3 and as shown in the first three rows, the RNN-V performs much better than the RNN-L on all the five datasets, which again indicates that employing pre-trained word vectors is better than learning the word vectors. The proposed HRNN outperforms the RNN-V and RNN-L models on all the five

Table 3. Performance (accuracy) of different models on different datasets. The top three results in each dataset are marked as bold. The symbol "-" means that no result is reported by the author on the corresponding dataset.

Model	MPQA	MR	Subj	CR
HRNN	**90.1**	**81.2**	**93.8**	**84.9**
RNN-V	88.7	80.4	92.8	83.4
RNN-L	85.4	73.1	90.7	74.6
CNN-static	**89.6**	81.0	93.0	**84.7**
CNN-non-static	**89.5**	**81.5**	93.4	84.3
CNN-multichannel	89.4	**81.1**	93.2	**85.0**
RAE	86.4	77.7	-	-
CCAE	87.2	77.8	-	-
Sent-Parser	86.3	79.5	-	-
NBSVM	86.3	79.4	93.2	81.8
MNB	86.3	79.0	**93.6**	80.0
G-dropout	86.1	79.0	93.4	82.1
F-dropout	86.3	79.1	**93.6**	81.9
Tree-CRF	86.1	77.3	-	81.4
CRF-PR	-	-	-	82.4

datasets, which again indicates the effectiveness of the added hidden layer. Comparing the RNN-based models with the CNN-based models shows that if the RNN model simply employs the pre-trained word vectors as input, its performance is worse than the CNN-based models on all the datasets.

However, after adding one hidden layer, our HRNN model performs slightly better than all the other baselines on the MPQA and Subj datasets and also achieves comparable results with the best performance on the MR and CR datasets.

The advantage of our model is more obvious on the MPQA dataset. This may result from the fact that the average sentence length in MPQA is much shorter than that in other datasets (shown in Table 2) and the syntactic parsing result on MPQA is more accurate than that on other datasets.

In conclusion, as shown in experiment 1 and experiment 2, by simply adding one hidden layer on the pre-trained word vectors, our model can improve the performance of the original RNN model.

6 Conclusion and Future Work

In this paper, we propose the hidden recursive neural network (HRNN) that can leverage the advantages of both using pre-trained word vectors and learning the word vectors, which are two common strategies used in the recursive neural network. Experiments on several sentence classification tasks show that using pre-trained word vectors is better than learning the word vectors and our model performs better than both of them.

We also investigate the effect of the label constraint of the internal nodes during training the RNN model and the experiment shows that the label constraint is better than no label constraint.

Since the performance of the RNN-based model relies on the pre-trained word vectors, we will investigate better ways to learn more expressive word vectors in the future. In addition, we will also investigate the effect of the syntactic parsing trees on the performance of the RNN-based model.

References

1. Baroni, M., Zamparelli, R.: Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 1183–1193 (2010)
2. Chen, X., Qiu, X., Zhu, C., Wu, S., Huang, X. J.: Sentence modeling with gated recursive neural network. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 793–798 (2015) doi: 10.18653/v1/D15-1092
3. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research, vol. 12, pp. 2493–2537 (2011)
4. Dong, L., Wei, F., Liu, S., Zhou, M., Xu, K.: A statistical parsing framework for sentiment classification. Computational Linguistics, vol. 41, no. 2, pp. 293–336 (2015) doi: 10.1162/COLI_a_00221
5. Dong, L., Wei, F., Zhou, M., Xu, K.: Adaptive multi-compositionality for recursive neural network models. IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 24, no. 3, pp. 422–431 (2014) doi: 10.1109/TASLP.2015.2509257
6. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, vol. 12, no. 7, pp. 2121–2159 (2011)
7. Goller, C., Kuchler, A.: Learning task-dependent distributed representations by backpropagation through structure. In: Proceedings of International Conference on Neural Networks (ICNN'96), vol. 1, pp. 347–352 (1996) doi: 10.1109/ICNN.1996.548916
8. Hermann, K. M., Blunsom, P.: The role of syntax in vector space models of compositional semantics. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, vol. 1, pp. 894–904 (2013)
9. 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
10. Irsoy, O., Cardie, C.: Deep recursive neural networks for compositionality in language. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, vol. 2, pp. 2096–2104 (2014)
11. Irsoy, O., Cardie, C.: Opinion mining with deep recurrent neural networks. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 720–728 (2014) doi: 10.3115/v1/D14-1080
12. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751 (2014) doi: 10.3115/v1/D14-1181
13. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature, vol. 521, pp. 436–444 (2015) doi: 10.1038/nature14539
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, vol. 2, pp. 3111–3119 (2013)

15. Mitchell, J., Lapata, M.: Vector-based models of semantic composition. In: Proceedings of ACL-08: HLT, pp. 236–244 (2008) https://aclanthology.org/P08-1028.pdf
16. Mitchell, J., Lapata, M.: Composition in distributional models of semantics. Cognitive Science, vol. 34, no. 8, pp. 1388–1429 (2010) doi: 10.1111/j.1551-6709.2010.01106.x
17. Mohammad, S.: Portable features for classifying emotional text. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 587–591 (2012)
18. Nakagawa, T., Inui, K., Kurohashi, S.: Dependency tree-based sentiment classification using CRFs with hidden variables. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 786–794 (2010)
19. Nguyen, T. H., Shirai, K.: PhraseRNN: Phrase recursive neural network for aspect-based sentiment analysis. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (2015) doi: 10.18653/v1/D15-1298
20. 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
21. Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, pp. 115–124 (2005) doi: 10.3115/1219840.1219855
22. Pennington, J., Socher, R., Manning, C. D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014) doi: 10.3115/v1/D14-1162
23. Qian, Q., Tian, B., Huang, M., Liu, Y., Zhu, X., Zhu, X.: Learning tag embeddings and tag-specific composition functions in recursive neural network. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 1365–1374 (2015) doi: 10.3115/v1/P15-1132
24. Silva, J., Coheur, L., Mendes, A. C., Wichert, A.: From symbolic to sub-symbolic information in question classification. Artificial Intelligence Review, vol. 35, no. 2, pp. 137–154 (2011) doi: 10.1007/s10462-010-9188-4
25. Socher, R., Huval, B., Manning, C. D., Ng, A. Y.: Semantic compositionality through recursive matrix-vector spaces. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 1201–1211 (2012)
26. Socher, R., Lin, C. C., Manning, C., Ng, A. Y.: Parsing natural scenes and natural language with recursive neural networks. In: Proceedings of the 28th International Conference on International Conference on Machine Learning, pp. 129–136 (2011)
27. Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., Manning, C. D.: Semi-supervised recursive autoencoders for predicting sentiment distributions. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 151–161 (2011)
28. Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1631–1642 (2013)
29. Tai, K. S., Socher, R., Manning, C. D.: Improved semantic representations from tree-structured long short-term memory networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (2015) doi: 10.3115/v1/P15-1150

30. Wang, S., Manning, C.: Fast dropout training. In: Proceedings of the 30th International Conference on Machine Learning, vol. 28, pp. 118–126 (2013)
31. Wang, S., Manning, C. D.: Baselines and bigrams: Simple, good sentiment and topic classification. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pp. 90–94 (2012)
32. Wiebe, J., Wilson, T., Cardie, C.: Annotating expressions of opinions and emotions in language. Language Resources and Evaluation, vol. 39, pp. 165–210 (2005) doi: 10.1007/s10579-005-7880-9
33. Yang, B., Cardie, C.: Context-aware learning for sentence-level sentiment analysis with posterior regularization. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pp. 325–335 (2014) http://aclweb.org/anthology/P14-1031

ISSN 1870-4069

Proposal for Modeling Brazilian Portuguese with Adaptive Grammars

Djalma Padovani, João José Neto

University of São Paulo,
School of Engineering of the University of São Paulo,
Brazil

djalma.padovani@usp.br, joao.jose@poli.usp.br

Abstract. Natural Language Processing uses different techniques for identifying elements of the language and the syntactic and semantic roles they carry out in the text under analysis. Traditionally, NLP systems are built with modules that divide the text, identify its elements, verify whether the syntactic trees are in accordance with grammar rules, and apply specific formalisms to validate the semantics. However, it is noticed that there are few formalisms that represent semantics in a syntactic way and such formalisms are either very complex or incomplete. Adaptive Grammars is a formalism in which a grammar can modify itself based on the character chain parsing and the application of rules associated to the context. It brings several advantages over similar techniques because it allows the representation of both syntactic and semantics together in a single model. This work presents a method for modeling Natural Languages using Adaptive Grammars and illustrates the proposal with an application to Brazilian Portuguese.

Keywords: Computational, terminology, formalisms and knowledge representation ·linguistics, grammars, automata, natural, language processing.

1 Introduction

Natural Language Processing uses several techniques for identifying elements of the language and the syntactic and semantic roles they carry out in the text in which they are inserted. Traditionally, NLP systems are built with modules that divide the text, identify its elements, verify whether the syntactic trees are in accordance with grammar rules, and apply specific formalisms to validate the semantics. However, it is noticed that there are few formalisms that represent semantics in a syntactic way and such formalisms are either very complex or incomplete. Adaptive technology presents a very practical formalism with great potential for application in all stages of natural language processing, including semantic validation. Adaptive Grammars is a formalism proposed by Iwai [1] in her doctoral thesis, in which grammatical rules are created dynamically from the processing of the input chain and from information about the context in which they are found.

That formalism presents several advantages over similar techniques, because with a single model it is possible to represent both syntactic and semantic characteristics. Iwai also demonstrates the computational equivalence between adaptive grammars and adaptive automata, which allows phrases of the language generated by an adaptive grammar to be recognized by adaptive automata. This work presents a method for modeling Natural Languages by using Adaptive Grammars and illustrates the proposal with an application to Brazilian Portuguese.

1.1 Concepts and Related Works

Natural Language Processing requires the development of programs that are capable of determining and interpreting the sentence structure at many levels of detail. Natural languages exhibit intricate structural behavior since the particular cases to be considered are profuse. Since natural languages are never formally designed, their syntactic rules are neither simple nor obvious and thus their computational processing is complex. Many methods are employed in NLP systems, adopting different paradigms, such as exact, approximate, pre-defined or interactive, intelligent or algorithmic methods [2].

Regardless of the method used, natural language processing involves the operations of lexical-morphological analysis, syntactic analysis, semantic analysis and pragmatic analysis [3]. The lexical-morphological analysis seeks to assign a morphological classification to each sentence word from the information stored in the lexicon [4]. The lexicon or dictionary is the data structure containing the lexical items and information corresponding to these items. Among the information associated with lexical items are the grammatical category of the item, such as noun, verb and adjective, and morphosyntactic-semantic values such as gender, number, grade, person, time, mode, verbal or nominal regency.

In the parsing step, the parser checks whether a sequence of words is a valid sentence in the language, recognizing it or not. The syntax analyzer makes use of a lexicon and a grammar that defines the rules of combining the items in sentence formation. In cases where there is a need to interpret the meaning of a text, the lexical-morphological analysis and the syntactic analysis are not enough, and it is necessary to perform a new type of operation, called semantic analysis [4]. The semantic analysis looks for mapping the syntactic structure to the domain of the application, making the structure gain a meaning.

The mapping is done by identifying the semantic properties of the lexicon and the semantic relationship between the items that compose it [5]. The pragmatic analysis seeks to reinterpret the structure that represents what was said to determine what was really meant. This category includes anaphoric relations, relations, determinations, focuses or themes, deictics and ellipses [6].

Adaptive grammar model is defined in [7] as a grammatical formalism that allows sets of production rules to be explicitly manipulated within a grammar. Types of manipulation include rule addition, deletion, and modification. The first description of grammar adaptivity (though not under that name) in the literature is generally [8, 9, 10] taken to be in a paper by Alfonso Caracciolo di Forino published in 1963 [11]. The next generally accepted reference to an adaptive formalism (extensible context-free grammars) came from Wegbreit in 1970 [12] in the study of extensible programming

languages, followed by the dynamic syntax of Hanford and Jones in 1973[13]. The work of Iwai in 2000 [1] takes the adaptive automata of Neto [14, 15] further by applying adaptive automata to context-sensitive grammars.

2 Adaptive Grammars

According to Iwai [1], adaptive grammar is a generative formalism capable of representing context-sensitive languages. What distinguishes this grammar from the conventional ones is its ability to self-modify as the sentences of language are derived. The modifications take place during the generation of the sentence, when applying production rules to which are associated adaptive actions, whose execution causes changes in the set of production rules and, possibly, in the set of non-terminal symbols.

A sentence ω belonging to the language represented by an adaptive grammar is generated from an original grammar G^0 and from a succession of intermediate G^1 ... G^{n-1} grammars, created whenever some adaptive action is activated during sentence generation, and finishes using G^n as the final grammar.

The author defines an adaptive grammar G as being a ordered triple (G^0, T, R^0), where: T is a finite, possibly empty, set of adaptive functions; $G^0 = (V_N^0, V_T, V_C, P_L^0, P_D^0, S)$ is an *initial grammar*, where V_N^0 is a finite non-empty set of *non-terminal symbols*, V_T is a finite non-empty set of *terminal symbols*, $V_N^0 \cap V_T = \varnothing$, V_C is a finite set of *context symbols*.

$V^0 = V_N^0 \cup V_T \cup V_C$, where V_N^0, V_T and V_C are disjoint sets two to two, $S \in V_N^0$ Is the initial symbol of grammar, P_L^0 is the set of rules of production applicable to situations free of context and P_D^0 is the set of rules of production applicable to situations dependent on context. The production rules consist of expressions with the following formats, i being an indicator of the number of adaptive changes already applied to the initial grammar:

Type1 or belonging to the set P_L^i, where i is an integer, greater than or equal to zero:

$N \rightarrow \{ A \} \alpha$, where $\alpha \in (V_T \cup V_N)^*$, $N \in V_N$ and A is an optional adaptive action associated with the production rule.

Type 2 or belonging to the set P_L^i, where i is an integer, greater than or equal to zero:

$N \rightarrow \varnothing$, where ∅ is a meta-symbol indicating the empty set.

This production indicates that although the non-terminal symbol N is defined, an empty set is derived, that is, there is no intended substitution for that non-terminal. This means that if this rule is applied in some derivation, the grammar will not generate any sentence. This rule is used for the case where there are rules that refer to non-terminals that should be dynamically defined, as a result of the application of some adaptive action.

Type 3 or belonging to the set P_D^i, where i is an integer, greater than or equal to zero:

$\alpha N \leftarrow \{ A \} \beta M$ where $\alpha \in V_C \cup \{\varepsilon\}$ e $\beta \in V_C$, or $\alpha N \rightarrow \{ A \} \beta M$, where $\alpha \in V_C$, $\beta \in V_T \cup \{\varepsilon\}$, N and $M \in V_N^i$, and A being an optional adaptive action. The first production has the arrow in reverse, indicating that β is being injected into the input chain. This production exchanges αN by βM, inserting context information. The second output has the arrow in the right direction, but has on its left side a context symbol followed by a non-terminal symbol, which indicates that αN is being replaced by βM

and generating β in the output chain. R^0 is a relationship of type (r, A), where $r \in (P_L{}^0 \cup P_D{}^0)$ and $A \in T$, $R^0 \subseteq P^0 \times (T \cup \{\varepsilon\})$. For each production rule r there is a relation R^0 that associates it to an adaptive action A.

A production rule to which an associated adaptive action is associated is called the adaptive production rule. The expression defining a production rule of type 1 of the adaptive grammar is of the context-free type less than the adaptive action, which, together with the productions in $P_D{}^0$, accounts for the representation of the context dependencies in this grammar.

According to Neto [14], adaptive actions are called adaptive functions. Iwai [1] uses the concept, originally designed for automata, and modifies it to be used in adaptive production rules. The notation proposed by the author is as follows:

```
Action name (list of formal parameters)
    { list of variables, list of generators:
            optional adaptive action
            elementary action 1,
            elementary action 2,
            .....
            elementary action n,
            optional adaptive action.
      }
```

In the adaptive function header there is the name of the adaptive function, followed by a list of formal parameters, separated by commas, in parentheses. When the adaptive function is called, this list will be filled with the corresponding values of the arguments passed in the particular function call, which values will be preserved intact throughout the function execution. The body of the adaptive function is represented by braces. It consists of a list of variables (separated by commas), a list of generators (separated by commas), followed by colon.

Next, the adaptive function may optionally contain the call of some adaptive function. This adaptive action is processed prior to the execution of the adaptive function being declared. Then, several elementary adaptive actions are listed, and in the end, there may be another adaptive function call to be processed after the execution of all the elementary actions of the adaptive function being declared.

Variables are symbols that give names to elements whose values are unknown at the time of the call of the adaptive function and which must be filled during the execution of the adaptive function as a result of adaptive query actions.

Generators are elements similar to variables, but are automatically filled at the beginning of the execution of the adaptive function, with values that do not repeat, preserving this value during the whole execution of the function.

According to the author's proposal, there are three types of elementary adaptive actions: query, addition and removal, which can be represented as follows, respectively:

? [N → { optional_adaptive_action } M],

+ [N → { optional_adaptive_action } M],

- [N → { optional_adaptive_action } M],

where $N \in V_N^i$, $M \in V^{i*}$, for some $i \in I$, where i represents the step of the evolution of grammar, and for some adaptive action, if it exists.

Elementary Consultation

This action verifies the existence of some production rule that presents the indicated format in the current set of productions of the grammar. When one or more symbols of the expression are represented by some variable, this action will fill those variables with the corresponding values that are found. Consequently, variables are filled as a result of performing the basic query actions. It should be noted that the variables, initially indefinite, are filled at most only once during the execution of an adaptive function.

Elementary Addition Action

This elementary action includes in the grammar a new production rule with the format indicated, being able to use variables and also generators for the definition of symbols that did not exist until the moment of its application. The addition of an existing rule is innocuous. The addition of rules that refer to undefined variables is also innocuous.

Elementary Removal Action

The execution of this elementary action is done in two steps. The first is to check the existence of the rule if you want to exclude with the consequent completion of the variables. If so, the removal is done, otherwise nothing is removed. It should be noted that, since there is more than one elementary adaptive action to be performed regardless of the order in which they were declared, the query rules take precedence. Between removals and additions, removals take precedence. The additions are always performed last. Elementary actions that refer to undefined variables will not be performed.

Adaptive Actions for Grammar Change

In the execution phase, adaptive actions are responsible for grammatical changes. When an adaptive action is performed, the grammar evolves, changing its non-terminal symbol sets and production rules.

Example of an adaptive function call:

$$\begin{aligned} A\,(A1, B1, C1) = \{&A2^*, B2^*, C2^*: \\ &+ [A1 \rightarrow \{A\,(A2, B2, C2)\,\}\ aA2] \\ &+ [A1 \rightarrow \{B\,(B2, C2)\}\ \varepsilon] \\ &+ [B \rightarrow b\ B2\ ] \\ &+ [C \rightarrow c\ C2]\}. \end{aligned}$$

Derivation Sequence

The generation of the sentences of a language represented by an adaptive grammar occurs through successive substitutions of the non-terminal symbols, according to the rules of grammar, starting from an initial symbol S. The primordial difference of this procedure is the presence of the adaptive actions that, when activated, alter the grammar that was being used until the moment the adaptive rule was applied. Another important difference with respect to conventional grammars refers to contextual productions, which can be used for context symbols in the derivation, so that the substitutions that

Table 1. Example of Celso Luft's phrasal patterns.

1	SS1	Vlig	SS2
2	SS	Vlig	Sadj
3	SS	Vlig	Sadv| Sadj

depend on them are performed only when said context symbol is explicitly present in the Sentential form when substitution of the non-terminal affected by it.

Normal form of adaptive grammars

The author also presents the concept of normal form of adaptive grammars and a standardization technique. Considering $G = (G^0, T, R^0)$ an adaptive grammar, where $G^0 = (V_N^0, V_T, V_C, P_L^0, P_D^0, S)$ the initial grammar that implements G, T is the set of adaptive actions, and R^0 the set of relationships that associates productions rules with adaptive actions.

The normalized grammar $G^N = (G^{0'}, T', R^{0'})$ is defined by $G^{0'}= (V_N^{0'}, V_T', V_C', P_L^{0'}, P_D^{0'}, S)$, the new grammar that implements G^N, T', its set of adaptive actions, and $R^{0'}$, the set of relationships that associates their production rules with the adaptive actions. If the grammar root S is recursive, a production $S' \rightarrow S$ is created, ensuring that all grammar roots are not self-recursive. Let's take a production rule with the following form:

$$A \rightarrow \{A\}\ a_1\ a_2 \ldots a_{n..}$$

a) if n = 0, then $A \rightarrow \varepsilon$, and the corresponding normal form is $A \rightarrow \{A\}\ \varepsilon$,

b) if n > 0, then the corresponding normal form is:

1. $A \rightarrow \{A'\}\ a_1\ A_{1,}$
2. $\alpha_1\ A_1 \rightarrow a_2\ A_2$

:

n. $\alpha_{n-1}\ A_{n-1} \rightarrow a_n\ A_n$

n+1. $\alpha_n\ A_n \leftarrow \alpha A_{n+1}$

n+2. $A_{n+1} \rightarrow \varepsilon$

with $\alpha \in V_C$, $\alpha_i \in V_C$ if $a_i \in V_N$, and $\alpha_i = \varepsilon$ if $\alpha_i \in V_T$.

A' is the normalized adaptive action equivalent to A. If $A = \varepsilon$, then $A' = \varepsilon$. If $a_1 \in V_N$, then the first rule of the set of normalized rules is changed to:

0. $A \rightarrow \{A'\}\ A_{0.}$

And the rule 1 becomes:

1. $A_0 \rightarrow a_1\ A_1$

If A is the initial symbol of the grammar, then there is not a rule n+1, because it is not necessary to insert a context symbol for the initial one. The latest normalized rules are now as follows:

n. $\alpha_{n-1}\ A_{n-1} \rightarrow a_n\ A_{n,}$

n+1. $\alpha_n\ A_n \rightarrow \varepsilon.$

The normalization of productions of type X→ ({A1} A1 | {A2} A2) is done as follows:

$X \rightarrow \{A_1'\}\ A_1X_1$ $\alpha_1X_1 \rightarrow X_2,$
$X_2 \rightarrow \{A_2'\}\ A_2X_3$ $\alpha_2X_3 \rightarrow X,$
$X_2 \rightarrow \varepsilon.$
with $\alpha_i \in V_C$ if $A_i \in V_N$, and $\alpha_i = \varepsilon$ if $A_i \in V_T$.

Ai' is the normalized adaptive action equivalent to Ai. If Ai = ε, then Ai'= ε. If each Ai is an expression (VN ∪ VT) * then apply the rules a and b before mentioned. The normalization of productions of the type X→ ({A1} A1 | {A2} A2 |... |{An} An) is done as follows:

$X \rightarrow \{\ A_1'\}\ A_1\ X_1$ $\alpha_1X_1 \rightarrow X_2,$
$X \rightarrow \{\ A_2'\}\ A_2\ X_1$ $\alpha_2X_1 \rightarrow X_2,$
: :
$X \rightarrow \{A_n'\}\ A_nX_1$ $\alpha_nX_1 \rightarrow X_2,$
$X_2 \rightarrow \varepsilon.$
with $\alpha_i \in V_C$ if $A_i \in V_N$, and $\alpha_i = \varepsilon$ if $A_i \in V_T.$

Ai' is the normalized adaptive action equivalent to Ai. If Ai = ε, then Ai'= ε. If each Ai is an expression (VN ∪ VT) * then apply the rules a and b before mentioned. The normalization of adaptive actions is done in a similar way to the previously presented production rules. But is these case, intermediary non-terminals are variables that will be filled as arguments are passed. For example, an adaptive action that contains elementary actions of consultation, removal and addition, and represented as follows:

A (x, y) = { ? [x→ a b c]
- [y→ d e]
+ [z → f] }.

In the normalized form becomes:

B (x, y) = { u_1, u_2, u_3, u_4, v_1, v_2, v_3, t_1, t_2:
? [x → a u_1], ?[u_4 → ε], -[v_3 → ε],
? [u_1 → b u_2], -[y→ d v_1], +[z → f t_1],
? [u_2 → c u_3], -[v_1 → e v_2], +[t_1← z' t_2],
? [u_3 ← x' u_4], -[v_2 ← y' v_3], +[t_2 → ε] }.

Context productions of the forms A ←αB, αA←βB, αA→B , and αA→ bB do not change, since they are defined only with the right side of the production, presenting at most only two elements.

3 Modern Brazilian Grammar of Celso Luft

The Modern Brazilian Grammar of Celso Luft [16] categorizes in a clear and precise way the different types of sentences of Portuguese Language, differing from the other grammars, which prioritize the description of the language to the detriment of the structural analysis of the same. Luft says that sentences are shaped by phrasal patterns, composed of elements called phrases. An example is given in Table 1.

Phrasal pattern is any immediate constituent of a sentence, and can play the role of subject, complement (direct and indirect object), predicative and adjunct adverbial. It is composed of one or more words, one being classified as a nucleus and the other as dependent. Dependent words may be located to the left or right of the nucleus. Luft uses the following names and abbreviations:

1. SS: Noun phrase - nucleus is a noun,
2. SV: Verb phrase - nucleus is a verb,
3. Sadj: Adjective phrase - nucleus is an adjective,
4. Sadv: Adverbial phrase - nucleus is an adverb,
5. SP: Prepositional phrase - formed by a preposition (Prep) plus one SS,
6. Vlig: linking verb,
7. Vi: intransitive verb,
8. Vtd: direct transitive verb,
9. Vti: indirect transitive verb,
10. Vtdi: direct and indirect transitive verb,
11. Vtpred: transitive verb predicative.

Luft also presents a comprehensive formula for sentence patterns:

[SS] V [SS] [SS | Sadj | Sadv | SP] [SP] [SP]

The symbol V represents all types of verbs: Vlig, Vi, Vtd, Vti, Vtdi, and Vtpred. Luft's formula allows us to generate any type of sentence pattern by simply using the productions defined by the grammar.

4 Introduction of Adaptive Mechanisms in Celso Luft's Grammar

The introduction of adaptive mechanisms in Celso Luft's Grammar has as main objective the use of a single model of representation that is able to generate sentences free and dependent on context.

Specifically, the adaptive characteristic of the Iwai Grammar will be used to generate the productions corresponding to the Luft sentence patterns in function of the lexical characteristics of the analyzed text. In the case of simple periods, verbs will be used to determine the pattern of previous and subsequent terms.

In the case of compound periods, conjunctions and relative pronouns will be used as defining elements of sentence break patterns. Therefore, the context dependence of this work is limited to the use of semantics to define syntactic sentence patterns and their respective rules of production. The work was divided into the following steps:

1. Consolidation of Celso Luft's Sentence Patterns.
2. Standardization of Production Rules.
3. Specification of Adaptive Grammar.

Table 2. Consolidated sentence patterns of Celso Luft's grammar.

Nominal Personal Patterns				
SS	Vlig	SS		
SS	Vlig	Sadj		
SS	Vlig	Sadv		
SS	Vlig	SP		
Verbal Personal Patterns				
SS	Vtd	SS		
SS	Vti	SP		
SS	Vti	Sadv		
SS	Vti	SP	SP	
SS	Vtdi	SS	SP	
SS	Vtdi	SS	Sadv	
SS	Vtdi	SS	SP	SP
SS	Vi			
Verbal Nominal Personal Patterns				
SS	Vtpred	SS	SS	
SS	Vtpred	SS	Sadj	
SS	Vtpred	SS	SP	

Table 2. Consolidated sentence patterns of Celso Luft's grammar – Continuation.

Verbal Nominal Personal Patterns				
SS	Vtpred	SS	Sadv	
SS	Vtpred	SS		
SS	Vtpred	Sadj		
SS	Vtpred	SP		
Impersonal Nominal Patterns				
	Vlig	SS		
	Vlig	Sadj		
	Vlig	Sadv		
	Vlig	SP		
Impersonal Verbal Patterns				
	Vtd	SS		
	Vti	SP		
	Vi			

4. Specification of Grammar Changes.

5. Specification of Semantics.

The first two steps are to prepare the Luft's Grammar so that it can be processed computationally. The third step corresponds to the presentation of the adaptive model itself. The fourth step presents the notation used to make changes in the grammar rules and the last step exemplifies the use of adaptive grammar in semantic analysis.

4.1 Consolidation of Celso Luft's Sentence Patterns

This proposal begins with the consolidation of the standards presented by Luft, unifying the types of patterns, in order to simplify computational processing. For example,

instead of using SS1 and SS2 to indicate noun phrases, only a single SS symbol will be used, which can be repeated according to the sentence pattern to which it belongs. The consolidated patterns are presented in Table 2.

4.2 Standardization of Production Rules

The production rules of the Celso Luft's Grammar have to be standardized to be used computationally. This work uses a model proposed Neto [17] with some modifications to refine the phrase structures. The productions are as follows:

S → [con] [PrRel] [SS] SV pont [S]

– S: Sentence

– PrRel: Relative pronoun

– con: Conjunction or relative pronoun,

– SS: Noun phrase,

– SV: Verb frase,

– pont: punctuation.

SS → [Sadj] SS [Sadj | SP | S] | S

– SS: Noun frase,

– Sadv: Adverbial frase,

– Sadj: Adjective frase,

– SP: Prepositional frase,

– S: Sentence.

SS → (([num | PrA] Sc) | Sp | PrPes)

– SS: Noun frase,

– num: numeral,

– PrA: Adjective pronoun,

– Sc: Common noun,

– Sp: Proper noun,

– PrPes: Personal Pronoun.

SV → [Neg] [Aux | PreV] V [SS | Sadj | Sadv | SP] | ((SS SP) | (SS Sadj) | (SP SP) | (SS SS) | (SS Sadv) | (SS SP SP))

– Neg: negation particle,

– Aux: Auxiliar passive voice particle,

– PreV: Pre-verbal,

– V: Vlig|Vtda|Vtdna|Vtdi, |Vtpred |Vti

– SS: Noun phrase,

– Sadv: Adverbial phrase,

– Sadj: Adjective phrase,

– SV: Verb frase,

– SP: Preposicional frase,

– SS: Noun phrase.

V → (Vlig | Vtda | Vtdna | Vtdi | Vtpred |Vti |Vi)

– V: Verb,

– Vlig: Linking verb,

– Vtda: Transitive verb convertible to passive voice,

– Vtdna: direct transitive verb not convertible to passive voice,

– Vtdi: Direct and indirect transitive verb,

– Vtpred: Predicative verb,

– Vti: Indirect transitive verb,

– Vi: Intransitive verb.

SP → Prep (SS | Sadj) | S

– SP: Preposicional phrase,

– Prep: preposition,

– SS: Noun phrase,

– Sadj: Adjective frase,

– S: Sentence.

Sadj → Sadj [SP] [S] | S

– Sadj: Sintagma adjetivo,

– SP: Sintagma preposicional,

– S: Sentença.

Sadj → [Adv] Adj [S] | S

– Sadj: Adjective frase,

– Adv: Adverb,

– Adj: Adjecive,

– S: Sentence.

Sadv → Sadv [SP] [S] | S

– Sadv: Adverbial frase,

– SP: Preposicional frase,

– S: Sentence.

Sadv → [Adv] Adv [S] | S

– Sadv: Adverbial frase,

– Adv: Adverb,

– S: Sentence.

PrA → [PrA] (Ind | ArtDef | ArtInd| Dem| Pos)

– PrA: Adjective pronoun,

– Ind: Indefinite pronoun,

– ArtDef: definite article,

– ArtInd: indefinite article,

– Dem: Indefinite demonstrative pronoun.

4.3 Specification of the Adaptive Grammar

The grammar is specified according to the production rules defined in item 4.2 and based on the Portuguese Language lexicon, being as follows:

$G = (G^0, T, R^0)$, where:
$G^0 = (V_N^0, V_T, V_C, P_L^0, P_D^0, S)$,
$V_N^0 = \{$ S, con, SS, SV, SS, Sadj, SP,num, PrA, Sc, Sp, PrPes, Neg, Aux, PreV,V, Vlig ,Vtda,Vtdna , Vtdi, Vtpred ,Vti ,Vi, Sadv, Prep, Ind, ArtDef, ArtInd, Dem, Pos, PrRel, Adv, Adj, pont $\}$,
V_T = {{lexicon},{,}, ; , . ,:,!,?,1, 2, 3, 4, 5, 6, 7,8, 9, 0},
P_L^0 = { S → [con] [PrRel] [SS] SV pont [S]
SS → [Sadj] SS [Sadj | SP] [S] | S
SS → (([num | PrA] Sc) | Sp | PrPes)
SV→ [Neg] [Aux | PreV] V [SS | Sadj | Sadv | SP] | ((SS SP) | (SS Sadj) | (SP SP) | (SS SS) | (SS Sadv) | (SS SP SP))
V → (Vlig | Vtda | Vtdna | Vtdi | Vtpred |Vti |Vi)
SP → Prep (SS | Sadj)
Sadj → Sadj [SP] [S] | S
Sadj → [Adv] Adj [S]

Sadv → Sadv [SP] [S] | S
Sadv → [Adv] Adv [S]
PrA → [PrA] (Ind | ArtDef | ArtInd| Dem| Pos)
con → (aditiva | adversativa | alternativa | conclusiva| explicativa |
integrante | causal | comparativa | concessiva | condicional |
conformativa | consecutiva | final | proporcional | temporal)[1]
num→ (0 |1 |2 | 3 |4 |5 |6 |7 |8 |9)|
(cardinal| ordinal| multiplicativo| fracionario)[2]
pont → (,| ; | . |: |! |?)
},
$V_C = \varnothing$,
$P_{D0} = \varnothing$.

With the normalization of the grammar G, the new grammar G'= (G^{0}', T', R^{0}') is defined as follows:

G^{0}' = ($V_N{}^{0}$', V_T', V_C', $P_L{}^{0}$', $P_D{}^{0}$', S),
$V_N{}^{0}$' = $V_N{}^{0}$, V_T' = V_T, V_C'= {δ , χ}.

$P_L{}^{0}$'∪ $P_D{}^{0}$' formed by the productions obtained with the normalization.

The normalization of the production S → [con|PrRel][SS] SV pont [S] generates:

1.^{0} SV → SV pont	5.^{0} S → SS S_1
2.^{0} S → SV	6.^{0} S1 → SV S
3.^{0} S → SV S	7.^{0} S → con S
4.^{0} S → SS SV	8.^{0} S → PrRel S

In this case, prior to the application of the Iwai standardization rules, is added the rule SV → SV pont, that incorporates the punctuation to the verbal phrase, since in the Grammar of Luft there is no production rules involving the punctuation.

The normalization of the production SS → [Sadj] SS [Sadj | SP | S] | S, generates:

9.^{0} SS → SS	13.^{0} SS → Sadj SS	17.^{0} SS2 → SS SP
10.^{0} SS → SS Sadj	14.^{0} SS → Sadj SS_1	18.^{0} SS→ Sadj SS_3
11.^{0} SS → SS SP	15.^{0} SS_1 → SS Sadj	19.^{0} SS3 → SS S
12.^{0} SS→ SS S	16.^{0} SS → Sadj SS_2	20.^{0} SS → S

The normalization of the production SS → (([num | PrA] Sc) | Sp | PrPes), generates:

21.^{0} SS → Sc
22.^{0} SS → Sp
23.^{0} SS → PrPes
24.^{0} SS→ num Sc
25.^{0} SS → PrA Sc

The normalization of the production SV→ [Neg] [Aux | PreV] V [SS | Sadj | Sadv | SP] | ((SS SP) | (SS Sadj) | (SP SP) | (SS SS) | (SS Sadv) | (SS SP SP)), generates:

[1] set of conjunction types of Portuguese Language
[2] set of numerical types of Portuguese Language

Simple Complements:

$26.^0$ SV → V	$40.^0$ SV → Aux V	$53.^0$ SV → Neg SV_7
$27.^0$ SV → V SS	$41.^0$ SV → Aux SV_1	$54.^0$ SV_7 → Aux SV_3
$28.^0$ SV → V Sadj	$42.^0$ SV → Aux SV_2	$55.^0$ SV → Neg
$29.^0$ SV → V Sadv	$43.^0$ SV → Aux SV_3	$56.^0$ SV_8 → Aux SV_4
$30.^0$ SV → V SP	$44.^0$ SV → Aux SV_4	$57.^0$ SV → Neg SV_9
$31.^0$ SV → Neg V	$45.^0$ SV → PreV V	$58.^0$ SV_9 → PreV V
$32.^0$ SV → Neg SV_1	$46.^0$ SV → PreV SV_1	$59.^0$ SV → Neg SV_{10}
$33.^0$ SV_1 → V SS	$47.^0$ SV → PreV SV_2	$60.^0$ SV_{10} → PreV SV_1
$34.^0$ SV → Neg SV_2	$48.^0$ SV → PreV SV_3	$61.^0$ SV → Neg SV_{11}
$35.^0$ SV_2 → V Sadj	$49.^0$ SV → PreV SV_4	$62.^0$ SV_{11} → PreV SV_2
$36.^0$ SV → Neg SV_3	$50.^0$ SV → Neg SV_5	$63.^0$ SV → Neg SV_{12}
$37.^0$ SV_3 → V Sadv	$51.^0$ SV_5 → Aux V	$64.^0$ SV_{12} → PreV SV_3
$38.^0$ SV → Neg SV_4	$51.^0$ SV → Neg SV_6	$65.^0$ SV → Neg SV_{13}
$39.^0$ SV_4 → V SP	$52.^0$ SV_6 → Aux SV_1	$66.^0$ SV_{13} → PreV SV_4

Compound Complements:

$67.^0$ SV → V (SS SP)	$85.^0$ SV → Aux SV_{14}	$103.^0$ SV → Neg SV_{23}
$68.^0$ SV → V (SS Sadj)	$86.^0$ SV → Aux SV_{15}	$104.^0$ SV23 → Aux SV_{17}
$69.^0$ SV → V (SP SP)	$87.^0$ SV → Aux SV_{16}	$105.^0$ SV → Neg SV_{24}
$70.^0$ SV → V (SS SS)	$88.^0$ SV → Aux SV_{17}	$106.^0$ SV_{24} → Aux SV_{18}
$71.^0$ SV → V (SS Sadv)	$89.^0$ SV → Aux SV_{18}	$107.^0$ SV → Neg SV_{25}
$72.^0$ SV → V (SS SP SP)	$90.^0$ SV → Aux SV_{19}	$108.^0$ SV_{25} → Aux SV_{19}
$73.^0$ SV → Neg SV_{14}	$91.^0$ SV → PreV SV_{14}	$109.^0$ SV → Neg SV_{26}
$74.^0$ SV_{14} → V (SS SP)	$92.^0$ SV → PreV SV_{15}	$110.^0$ SV_{26} → PreV SV_{14}
$75.^0$ SV → Neg SV_{15}	$93.^0$ SV → PreV SV_{16}	$111.^0$ SV → Neg SV_{27}
$76.^0$ SV_{15} → V (SS Sadj)	$94.^0$ SV → PreV SV_{17}	$112.^0$ SV_{27} → PreV SV_{15}
$77.^0$ SV → Neg SV_{16}	$95.^0$ SV → PreV SV_{18}	$113.^0$ SV → Neg SV_{28}
$78.^0$ SV_{16} → V (SP SP)	$96.^0$ SV → PreV SV_{19}	$114.^0$ SV_{28} → PreV SV_{16}
$79.^0$ SV → Neg SV_{17}	$97.^0$ SV → Neg SV_{20}	$115.^0$ SV → Neg SV_{29}
$80.^0$ SV_{17} → V (SS SS)	$98.^0$ SV_{20} → Aux SV_{14}	$116.^0$ SV_{29} → PreV SV_{17}
$81.^0$ SV → Neg SV_{18}	$99.^0$ SV → Neg SV_{21}	$117.^0$ SV → Neg SV_{30}
$82.^0$ SV_{18} → V (SS Sadv)	$100.^0$ SV_{21} → Aux SV_{15}	$118.^0$ SV_{30} → PreV SV_{18}
$83.^0$ SV → Neg SV_{19}	$101.^0$ SV → Neg SV_{22}	$119.^0$ SV → Neg SV_{31}
$84.^0$ SV_{19} → V (SS SP SP)	$102.^0$ SV_{22} → Aux SV_{16}	$120.^0$ SV_{31} → PreV SV_{19}

The normalization of the production SP → Prep (SS | Sadj), generates:

$121.^0$ SP → Prep SS
$122.^0$ SP → Prep Sadj

The normalization of the production Sadj → Sadj [SP] [S] | S, generates:

$123.^0$ Sadj → Sadj
$124.^0$ Sadj → Sadj SP
$125.^0$ Sadj → Sadj S
$126.^0$ Sadj → Sadj $Sadj_1$
$127.^0$ $Sadj_1$ → SP S
$128.^0$ Sadj → S

The normalization of the production Sadj → [Adv] Adj [S] , generates:

129.^{0} Sadj → Adj
130.^{0} Sadj → Adv Adj
131.^{0} Sadj → Adv $Sadj_2$
132.^{0} $Sadj_2$ → Adj S
133.^{0} Sadj → Adj S

The normalization of the production Sadv → Sadv [SP] [S] | S, generates:

134.^{0} Sadv → Sadv
135.^{0} Sadv → Sadv SP
136.^{0} Sadv → Sadv $Sadv_1$
137.^{0} $Sadv_1$ → SP S
138.^{0} Sadv → Sadv S
139.^{0} Sadv → S

The normalization of the production Sadv → [Adv] Adv [S], generates:

140.^{0} Sadv → Adv
141.^{0} Sadv → Adv Adv
142.^{0} Sadv → Adv $Sadv_2$
143.^{0} $Sadv_2$ → Adv S

The normalization of the production V → ({A(t)}Vlig | {A(t)}Vtda | {A(t)}Vtdna | {A(t)}Vtdi | {A(t)}Vtpred |{A(t)}Vti|{A(t)}Vi), generates:

44.^{0} V →{A(Vlig)}V	148.^{0} V → {A(Vtpred)}V
145.^{0} V → {A(Vtda)}V	149.^{0} V →{A(Vti)}V
146.^{0} V → {A(Vtdna)}V	150.^{0} V → {A(Vi)}V
147.^{0} V → {A(Vtdi)}V	

The normalization of the production PrA → [PrA] (Ind | ArtDef | ArtInd| Dem| Pos), generates:

151.^{0} PrA → Ind	156.^{0} PrA → PrA Ind
152.^{0} PrA → ArtDef	157.^{0} PrA → PrA ArtDef
153.^{0} PrA → ArtInd	158.^{0} PrA → PrA ArtInd
154.^{0} PrA → Dem	159.^{0} PrA → PrA Dem
155.^{0} PrA → Pos	160.^{0} PrA → PrA Pos

The normalization of the production con → (aditiva | adversativa | alternativa | conclusiva| explicativa|integrante | causal | comparativa | concessiva | condicional | conformativa | consecutiva | final | proporcional | temporal), generates:

con →{B(cAditiva)}con	168.^{0} con → {B(sConcessiva)} con
162.^{0} con → {B(cAdversativa)} con	169.^{0} con → {B(sCondicional)} con
163.^{0} con →{B(cAalternativa)} con	170.^{0} con → {B(sConformativa)} con
164.^{0} con → {B(cExplicativa)} con	171.^{0} con → {B(sConsecutiva)} con
165.^{0} con → {B(sIntegrante)} con	172.^{0} con → {B(sFinal)} con
166.^{0} con → {B(sCausal)} con	173.^{0} con → {B(sProporcional)} con
167.^{0} con → {B(sComparativa)} con	174.^{0} con → {B(sTemporal)} con

The normalization of the production num→ (0 |1 |2 | 3 |4 |5 |6 |7 |8 |9)| (cardinal | ordinal | multiplicativo | fracionario), generates:

175.^{0} num → 0	182.^{0} num → 5
176.^{0} num → 1	183.^{0} num → 6
177.^{0} num → 2	184.^{0} num → 7
178.^{0} num → 3	185.^{0} num → 8
179.^{0} num → 4	186.^{0} num → 9
180.^{0} num → cardinal	187.^{0} num → ordinal
181.^{0} num → multiplicativo	188.^{0} num → fracionario

The normalization of the production pont → (,| ; | . |: |! |?), generates:

189.^{0} pont → ,	192.^{0} pont → :
190.^{0} pont → ;	193.^{0} pont → !
191.^{0} pont → .	194.^{0} pont → ?

The normalization of the production PrRel → (substantivo | adjetivo| adverbio), generates:

195.^{0} PrRel → {C(PrRelSubs)} PrRel
196.^{0} PrRel → {C(PrRelAdj)} PrRel
197.^{0} PrRel → {C(PrRelAdv)} PrRel

The other non-terminals generate production rules based on the classification of the Portuguese Language lexicon.

4.4 Specification of Grammar Changes

Adaptive actions are used to modify the rules of production according to the context defined by the type of verb found in the text and by the presence of conjunctions or relative pronouns. However, for these modifications to be made, it is necessary to exclude and include a very large number of production rules. In view of this need, Iwai's notation has been extended with the instruction R, which means replace, i.e. exchange of a set of rules from the previous grammar by a new set of rules.
The syntax of the R statement is as follows:

$$R: <_G{}^n> <P_{I\text{-}F}> : <_G{}^{n+1}> P',$$

where:

R: replace

$_G{}^n$: Grammar prior to the update

$P_{I\text{-}F}$: Production rules for updating, where I = Interval start and F = interval end. It is allowed to add more than one rule set by using the comma for separating

$_G{}^{n+1}$: Grammar posterior to the update

P': new production rule

∅ : Symbol indicating that the rules set out in $P_{I\text{-}F}$ will be removed

The following examples show the use of R instruction in an adaptive action A.

$$T = \{ \quad A\ (t=Vlig) = F\ (t) = \{ \quad + [R: {}_G{}^n P_{26\text{-}66:}\ {}_G{}^{n+1} V \leftarrow \chi V]$$
$$+ [{}_{G\,n+1}\chi V \rightarrow Vlig]$$
$$+ [R: {}_G{}^n P_{67\text{-}120:}\ {}_G{}^{n+1} \varnothing],$$

$$T = \{ \quad B\ (p = cAditiva) = B\ (p) = \{ \quad + [R: {}_G{}^n P_7:\ {}_G{}^{n+1}\ con \leftarrow \delta con]$$
$$+ [G\ n+1\ \delta con \rightarrow cAditiva]$$
$$+ [R: {}_G{}^n P_7:\ {}_G{}^{n+1} \varnothing].$$

4.5 Semantics

The formalism presented by Iwai provides support to include additional features to solve problems where context is important for selection of production rules. For example, the notation allows you to check whether the analyzed sentence is correct or to resolve possible ambiguities if you have information about the type of verb, time, mode and person of the analyzed verb. In the example below, the adaptive action A uses as input parameters the type identification of the verb, mode and person to define the adaptive functions:

$$T = \{ \quad A\ (verb=Vlig, time = Present, mode = Singular, person = First) =$$
$$F\ (verb, time, mode, person) = \{ \quad + [R: {}_G{}^n P_{i\text{-}f:}\ {}_G{}^{n+1} V \leftarrow \chi V$$
$$+ [{}_G{}^{n+1}\chi V \rightarrow V_{lig,\ Present,\ Singular,\ First}]$$
$$+ [R: {}_G{}^n P_{i\text{-}f:}\ {}_G{}^{n+1} \varnothing] \qquad \}.$$

Adaptive formalism was used to add contextual features to grammar, which would not be possible using only context-free grammar. Analogously, it is possible to include rules for analysis of nominal agreement or any other type of semantic content, without the need to use any element other than formalism. Adaptive grammars can also be used to represent the use of probabilistic information in the selection of production rules. Such a technique is used when there is more than one applicable production rule and there is insufficient syntactic and semantic information to disambiguate them.

In this case, it is possible to use the probability of occurrence of the rules as a choice factor and the formalism of Iwai can be used to represent this type of contextual information. In the example below, the adaptive action A uses as input parameter probability of occurrence of the evaluated production rules and an indication to use the rule of maximum probability:

$$SV_{prob} \rightarrow V\ SP\ 10\%,$$
$$SV_{prob} \rightarrow V\ (SS\ SP)\ 90\%,$$
$$T = \{ \quad A\ (t=SV, u = prob, v=max) =$$
$$F\ (SV, prob, max) = \{ + [R: {}_G{}^n P_{inicio\text{-}fim}:\ {}_G{}^{n+1} SV \leftarrow \chi SV_{prob}]$$
$$+ [{}_G{}^{n+1}\chi SV_{prob} \rightarrow \chi SV_{max}]$$
$$? [{}_G{}^{n+1}\chi SV_{max}]$$
$$+ [{}_G{}^{n+1}\chi SV_{max} \rightarrow V\ (SS\ SP)]$$
$$+ [R: {}_G{}^n P_{inicio\text{-}fim}:\ {}_G{}^{n+1} \varnothing]. \qquad \}.$$

5 Conclusions

Adaptivity extends the capabilities of conventional grammars, providing syntactic and semantic representation power in a single device. The formality presented by Margarete Iwai structures this knowledge and allows it to be applied to natural language grammars. This work presented a method for modeling natural languages using Adaptive Grammars and illustrates the proposal with an application to Brazilian Portuguese, based on Modern Brazilian Grammar of Celso Luft. Luft's production patterns were formatted in the Iwai model and adaptive actions were created for the generation of coordinated and subordinate sentences, with validation of Luft's proposed sentence patterns as a function of the context of the analysis.

Due to the amount of production rules used by the Luft grammar, a new adaptive action, the R instruction, was introduced, replacing a set of rules from the previous grammar with a new set of rules, allowing a large set of production rules to be updated with a single instruction. The semantics representation was exemplified with an ambiguity problem, in which context was important to choose the most adequate production. Finally, the robustness of the model was proven with an example in which it was included probabilistic information in the selection of production rules.

6 Future Works

The intention is to continue this research through the development of several aspects not considered in the scope of this work. An example is the syntactic patterns analyzed before the verb of the sentence, which can be changed the moment the type of verb is identified. Another example is the verification of the possibility of inversion of the sentence patterns, also depending on the type of the verb. Finally, it is intended to evaluate the application of nominal and verbal regency criteria to verify the adequacy of the sentence analyzed to the cult pattern of the Portuguese Language.

References

1. Iwai, M. K.: A gramatical formulation for context-dependent languages. Escola Politécnica da Universidade de São Paulo. Tese de Doutorado (2000)
2. Neto, J. J., Moraes, M. D.: Using adaptive formalisms to describe context-dependencies in natural language. In: International Workshop on Computational Processing of the Portuguese Language, Springer, vol. 2721, pp. 94–97 (2003) doi: 10.1007/3-540-45011-4_14
3. Rich, E., Knight, K., Shivashankar, B. N.: Artificial intelligence (chapter natural language processing), 3rd Edition, Tata McGraw-Hill, (2009)
4. Vieira, R., Lima, V. L. S.: Linguística computacional: princípios e aplicações. IX Escola de Informática da SBC–Sul (2001)
5. Fuchs, C., Le-Goffic, P.: Initiation aux problèmes des linguistiques contemporaines. Hachette Université (1975)
6. Nunes, M. D. G. V., Pardo, T. A.: Introdução ao Processamento das Línguas Naturais. Notas didáticas do ICMC, Universidade São Paulo, vol. 180 (1999)
7. Shutt, J. N.: What is an Adaptive Grammar? (2001)
8. Christiansen, H.: A survey of adaptable grammars. ACM SIGPLAN Notices, vol. 25, no. 11, pp. 35–44 (1990)

9. Shutt, J. N.: Recursive adaptable grammars. Master's Thesis, Worcester Polytechnic Institute (1993)
10. Jackson, Q. T.: Adapting to babel: adaptivity and context-sensitivity in parsing. Ibis Publications (2006)
11. Di-Forino, A. C.: Some remarks on the syntax of symbolic programming languages. Communications of the ACM, vol. 6, no. 8, pp. 456–460 (1963)
12. Wegbreit, B.: Studies in extensible programming languages, ESD-TR-70-297, Harvard University Cambridge, Massachusetts (1970)
13. Hanford, K. V., Jones, C. B.: Dynamic syntax: A concept for the definition of the syntax of programming languages. Annual Review in Automatic Programming, Pergamon Press, pp. 115–142 (1973)
14. Neto, J. J.: Adaptive automata for context-sensitive languages. ACM SIGPLAN Notices, vol. 29, no. 9, pp. 115–124 (1994)
15. Neto, J. J.: Adaptive rule-driven devices - general formulation and case study. Lecture Notes in Computer Science. In: Implementation and Application of Automata 6th International Conference, CIAA'01, Springer, vol. 2494, pp. 234–250 (2001)
16. Luft, C.: Moderna gramática brasileira. 2ª. Edição Revista e Atualizada, Editora Globo (2002)
17. Neto, J. J.: Adaptive handling of some linguistic phenomena. Available at: https://sites.google.com/site/2015pcs5004/material-para-download

ISSN 1870-4069

Extractive Summarization Using Deep Learning

Sukriti Verma, Vagisha Nidhi

Delhi Technological University,
India

sukriti_bt2k14@dtu.ac.in,
vagisha.nda@gmail.com

Abstract. This paper proposes a text summarization approach for factual reports using a deep learning model. This approach consists of three phases: feature extraction, feature enhancement, and summary generation, which work together to assimilate core information and generate a coherent, understandable summary. We have extracted various features from each sentence, and are using a Restricted Boltzmann Machine to enhance and abstract those features to improve resultant accuracy without losing any important information. The sentences are scored using those enhanced features and an extractive summary is constructed. Experimentation carried out on several articles demonstrates the effectiveness of the proposed approach[1].

Keywords: Unsupervised, single document, deep learning, extractive.

1 Introduction

A summary can be defined as a text produced from one or more texts, containing a significant portion of the information from the original text(s), and that is no longer than half of the original text(s) [7]. According to [11], text summarization is the process of distilling the most important information from a source to produce an abridged version for a particular user and task(s). When this is done by means of a computer, i.e. automatically, we call it Automatic Text Summarization. This process can be seen as a form of compression and it necessarily suffers from information loss but it is essential to tackle the information overload due to abundance of textual material available on the Internet.

Text Summarization can be classified into extractive summarization and abstractive summarization based on the summary generated. Extractive summarization is creating a summary based on strictly what you get in the original text. Abstractive summarization mimics the process of paraphrasing a text. Text(s) summarized using this technique looks more human-like and produces condensed summaries. These techniques are much harder to implement than the extractive summarization techniques. In this paper, we follow the extractive methodology to develop techniques for summarization of factual reports or descriptions.

[1]The source code for this paper is available at https://github.com/vagisha-nidhi/TextSummarizer

We have developed an approach for single-document summarization using deep learning. So this paper intends to propose an approach by referencing the architecture of the human brain. It is broken down into three phases: feature extraction [4], feature enhancement, and summary generation based on values of those features.

Since it can be very difficult to construct high-level, abstract features from raw data, we use deep learning in the second phase to build complex features out of simpler features extracted in the first phase. These extracted features depend highly on how factual the given document is. In the end, we have run the proposed algorithm on several factual reports to evaluate and demonstrate the effectiveness of the proposed approach based on the measures such as Recall, Precision, and F-measure.

2 Related Works

Most early work on text summarization was focused on technical documents and early studies on summarization aimed at summarizing from pre-given documents without any other requirements, which is usually known as generic summarization [2]. Luhn [10] proposed that the frequency of a particular word in an article provides a useful measure of its significance. A number of key ideas, such as stemming and stop word filtering, were put forward in this paper that have now been understood as universal preprocessing steps to text analysis.

Baxendale [1] examined 200 paragraphs and found that in 85% of the paragraphs, the topic sentence came as the first one and in 7% of the time, it was the last sentence. This positional feature has been used in many complex machine learning based systems since. Edmundson [6] focused his work around the importance of word frequency and positional importance as features.

Two other features were also used: cue words, and the skeleton structure of the document. Weights were associated with these features manually and finally sentences were scored. During evaluation, it was found that around 44% of the system generated summaries matched the target summaries written manually by humans.

Upcoming researchers in text summarization have approached it problem from many aspects such as natural language processing [19], statistical modelling [5] and machine learning. While initially most machine learning systems assumed feature independence and relied on naive-Bayes methods, other recent ones have shifted focus to selection of appropriate features and learning algorithms that make no independence assumptions. More recent papers, in contrast, used neural networks towards this goal.

Text Summarization can be done for one document, known as single-document summarization [17], or for multiple documents, known as multi-document summarization [15]. On basis of the writing style of the final summary generated, text summarization techniques can be divided into extractive methodology and abstractive methodology [18]. The objective of generating summaries via the extractive approach is choosing certain appropriate sentences as per the requirement of a user.

Due to the idiosyncrasies of human-invented languages and grammar, extractive approaches, which select a subset of sentences from the input documents to form a summary instead of paraphrasing like a human [3], are the mainstream in the area.

Almost all extractive summarization methods have three main obstacles. The first obstacle is the ranking problem, i.e. how you rank words, phrases and/or sentences. The second obstacle is the selection problem, i.e. how to select a subset of those ranked units [8]. The third obstacle is the coherence problem, i.e. how to ensure that the selected units form an understandable summary rather than being a set of disconnected words, phrases and/or sentences.

Algorithms that determine the relevance of a textual unit, that is words, phrases and/or sentences, with respect to the requirement of the user are used to solve the ranking problem. The selection and coherence problems are solved by methods that improve diversity, minimize redundancy and pick up phrases and/or sentences that are somewhat similar so that more relevant information can be covered by the summary in lesser words and the summary is coherent. Our approach solves the ranking problem by learning a certain set of features for each sentence.

On the basis of these features, a score is calculated for each sentence and sentences are arranged in decreasing order of their scores [16]. Even with a list of ranked sentences, it is not a trivial problem to select a subset of sentences for a coherent summary which includes diverse information, minimizes redundancy and is within a word limit. Our approach solves this problem as follows.

The most relevant sentence is the first sentence in this sorted list and is chosen as part of the subset of sentences which will form the summary. Then the next sentence selected is a sentence having highest Jaccard similarity with the first sentence and is picked from the top half of the list. This process is recursively and incrementally repeated to select more sentences until limit is reached.

3 Proposed Approach

3.1 Preprocessing

Preprocessing is crucial when it comes to processing text. Ambiguities can be caused by various verb forms of a single word, different accepted spellings of a certain word, plural and singular terms of the same things. Moreover, words like a, an, the, is, of etc. are known as stop words. These are certain high frequency words that do not carry any information and don't serve any purpose towards our goal of summarization. In this phase we do:

1. **Document Segmentation:** The text is divided into paragraphs so as to keep a track of which paragraph each sentence belongs to and what is the position of a sentence in its respective paragraph.
2. **Paragraph Segmentation:** The paragraphs are further divided into sentences.
3. **Word Normalization:** Each sentence is broken down into words and the words are normalized. Normalization involves lemmatization and results in all words being in one common verb form, crudely stemmed down to their roots with all ambiguities removed. For this purpose, we use Porters algorithm.
4. **Stop Word Filtering:** Each token is analyzed to remove high frequency stop words.

5. **PoS Tagging:** Remaining tokens are Part-of-Speech tagged into verb, noun, adjective etc. using the PoS Tagging module supplied by NLTK [12].

3.2 Feature Extraction

Once the complexity has been reduced and ambiguities have been removed, the document is structured into a sentence-feature matrix. A feature vector is extracted for each sentence. These feature vectors make up the matrix. We have experimented with various features. The combination of the following 9 sentence features has turned out most suitable to summarize factual reports. These computations are done on the text obtained after the preprocessing phase:

1. **Number of thematic words:** The 10 most frequently occurring words of the text are found. These are thematic words. For each sentence, the ratio of no. of thematic words to total words is calculated:

$$\text{Sentence\_Thematic} = \frac{\text{No. of thematic words}}{\text{Total words}}. \tag{1}$$

2. **Sentence position:** This feature is calculated as follows:

$$\text{Sentence\_Position} = \begin{cases} 1, \text{ if its the first or last sentence of the text,} \\ \cos((\text{SenPos} - \text{min})((1/\text{max}) - \text{min})), \text{ otherwise,} \end{cases} \tag{2}$$

where, SenPos = position of sentence in the text, min = th x N, max = th x 2 x N, N is total number of sentences in document, th is threshold calculated as 0.2 x N.

By this, we get a high feature value towards the beginning and ending of the document, and a progressively decremented value towards the middle.

3. **Sentence length:** This feature is used to exclude sentences that are too short as those sentences will not be able to convey much information:

$$\text{Sentence\_Length} = \begin{cases} 0, \text{ if number of words is less than 3,} \\ \text{No. of words in the sentence, otherwise.} \end{cases} \tag{3}$$

4. **Sentence position relative to paragraph:** This comes directly from the observation that at the start of each paragraph, a new discussion is begun and at the end of each paragraph, we have a conclusive closing:

$$\text{Position\_In\_Para} = \begin{cases} 1, \text{ if it is the first or last sentence of a paragraph,} \\ 0, \text{ otherwise.} \end{cases} \tag{4}$$

5. **Number of proper nouns:** This feature is used to give importance to sentences having a substantial number of proper nouns. Here, we count the total number of words that have been PoS tagged as proper nouns for each sentence.

6. **Number of numerals:** Since figures are always crucial to presenting facts, this feature gives importance to sentences having certain figures. For each sentence we calculate the ratio of numerals to total number of words in the sentence:

$$\text{Sentence\_Numerals} = \frac{\text{No. of numerals}}{\text{Total words}}. \tag{5}$$

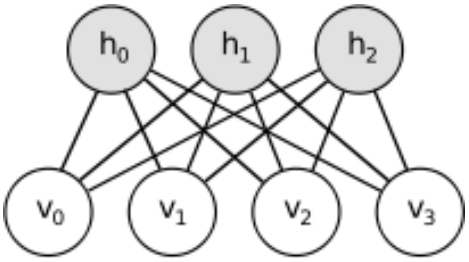


Fig. 1. A restricted Boltzmann machine [9].

7. **Number of named entities:** Here, we count the total number of named entities in each sentence. Sentences having references to named entities like a company, a group of people etc. are often quite important to make any sense of a factual report.
8. **Term Frequency-Inverse Sentence Frequency (TF – ISF):** Since we are working with a single document, we have taken TF-ISF feature into account rather than TF-IDF. Frequency of each word in a particular sentence is multiplied by the total number of occurrences of that word in all the other sentences. We calculate this product and add it over all words:

$$\mathrm{TF-ISF} = \frac{\log(\sum_{\text{all words}} \mathrm{TF} * \mathrm{ISF})}{\text{Total words}}. \tag{6}$$

9. **Sentence to Centroid similarity:** Sentence having the highest TF-ISF score is considered as the centroid sentence. Then, we calculate cosine similarity of each sentence with that centroid sentence:

$$\mathrm{Sentence\_Similarity} = \mathrm{cosine\_sim}(\text{sentence}, \text{centroid}). \tag{7}$$

At the end of this phase, we have a sentence-feature matrix.

3.3 Feature Enhancement

The sentence-feature matrix has been generated with each sentence having 9 feature vector values. After this, recalculation is done on this matrix to enhance and abstract the feature vectors, so as to build complex features out of simple ones. This step improves the quality of the summary.

To enhance and abstract, the sentence-feature matrix is given as input to a Restricted Boltzmann Machine (RBM) which has one hidden layer and one visible layer. A single hidden layers will suffice for the learning process based on the size of our training data. The RBM that we are using has 9 perceptrons in each layer with a learning rate of 0.1.

We use Persistent Contrastive Divergence method to sample during the learning process [9]. We have trained the RBM for 5 epochs with a batch size of 4 and 4 parallel Gibbs Chains, used for sampling using Persistent CD method. Each sentence feature vector is passed through the hidden layer in which feature vector values for each sentence are multiplied by learned weights and a bias value is added to all the feature vector values which is also learned by the RBM.

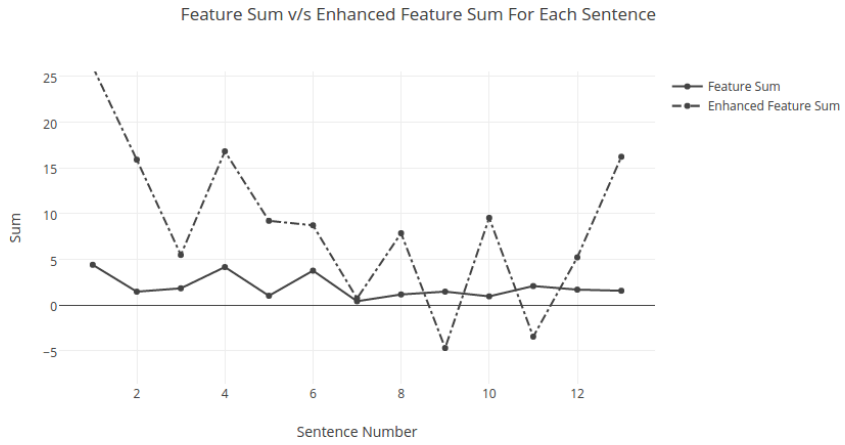


Fig. 2. Comparison between feature vector sum and enhanced feature vector sum.

At the end, we have a refined and enhanced matrix. Note that the RBM will have to be trained for each new document that has to be summarized. The idea is that no document can be summarized without going over it. Since each document is unique in the features extracted in section 3.2, the RBM will have to be freshly trained for each new document.

3.4 Summary Generation

The enhanced feature vector values are summed to generate a score against each sentence. The sentences are then sorted according to decreasing score value. The most relevant sentence is the first sentence in this sorted list and is chosen as part of the subset of sentences which will form the summary.

Then the next sentence we select is the sentence having highest Jaccard similarity with the first sentence, selected strictly from the top half of the sorted list. This process is recursively and incrementally repeated to select more sentences until a user-specified summary limit is reached. The sentences are then re-arranged in the order of appearance in the original text. This produces a coherent summary rather than a set of haywire sentences.

4 Results and Performance Evaluation

Several factual reports from various domains of health, technology, news, sports etc. with varying number of sentences were used for experimentation and evaluation. The proposed algorithm was run on each of those and system-generated summaries were compared to the summaries produced by humans. Feature Extraction and Enhancement is carried out as proposed in sections 3.2 and 3.3 for all documents. The values of feature vector sum and enhanced feature vector sum for each sentence of one such document have been plotted in Fig 2.

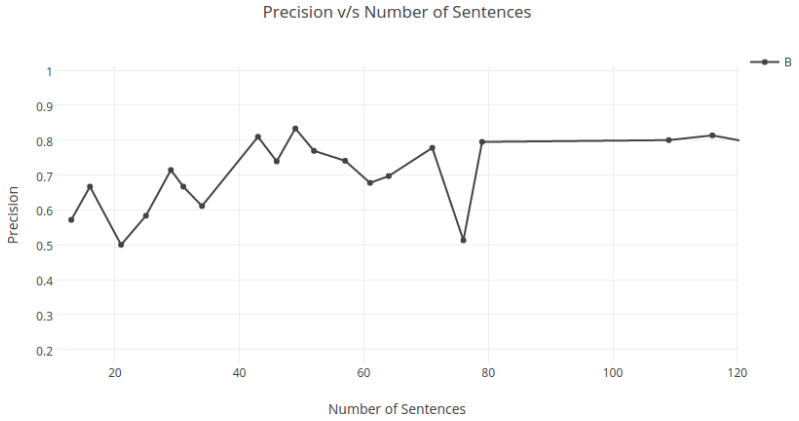


Fig. 3. Precision values corresponding to summaries of various documents.

The Restricted Boltzmann Machine has extracted a hierarchical representation out of data that initially did not have much variation, hence discovering the latent factors. The sentences have then been ranked on the basis of final feature vector sum and summaries are generated as proposed in section 3.4.

Evaluation of the system-generated summaries is done based on three basic measures: Precision, Recall and F-Measure [14]. It can be seen that as the number of sentences in the original document cross a certain threshold, the Restricted Boltzmann Machine has ample data to be trained successfully and summaries with high precision and recall values are generated. See Fig 3 and 4.

F-Measure is defined as follows [13]:

$$\text{F-Measure} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}. \tag{8}$$

5 Comparative Analysis

The existing approach was executed for the same set of articles with just one layer of RBM, rather than two as it specifies and average values of Precision, Recall and F-Measure were plotted for drawing a comparison between the existing approach and the proposed approach, while keeping the amount of computation constant.

The proposed approach has an average precision value of 0.7 and average recall value of 0.63 which are both higher than those of the existing approach. Hence, the proposed approach responds better for summarization of factual reports.

6 Conclusion

We have developed an algorithm to summarize single-document factual reports. The algorithm runs separately for each input document, instead of learning rules from a corpus, as each document is unique in itself. This is an advantage that our approach provides.

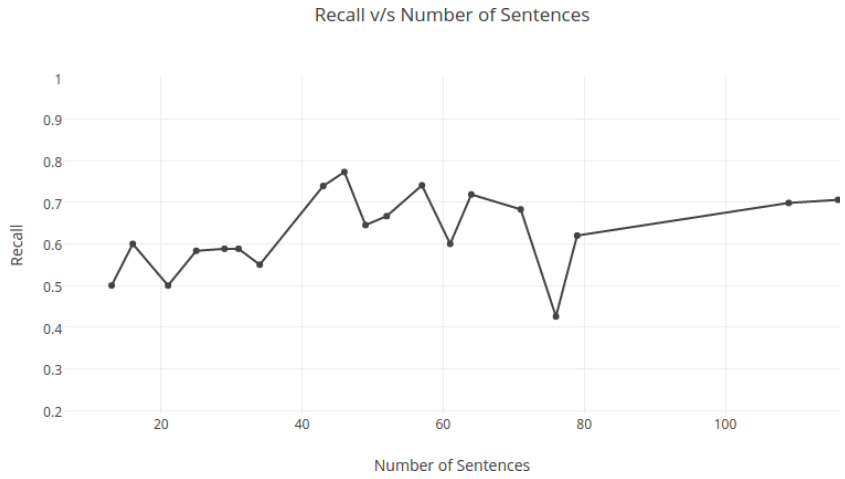


Fig. 4. Recall values corresponding to summaries of various documents.

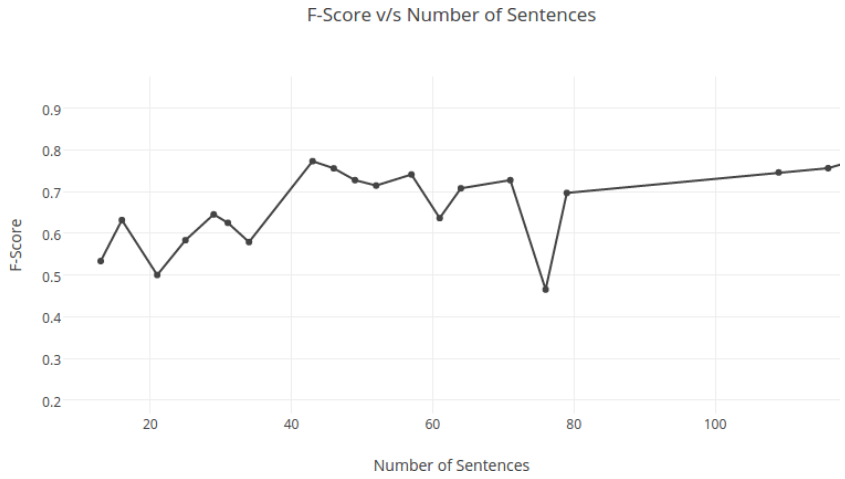


Fig. 5. F-Measure values corresponding to summaries of various documents.

We extract 9 features from the given document and enhance them to score each sentence. Recent approaches have been using 2 RBMs stacked on top of each other for feature enhancement. Our approach uses only one RBM and, works effectively and efficiently for factual reports.

This has been demonstrated by hand-picking factual descriptions from several domains and comparing the system-generated summaries to those written by humans. This approach can further be developed by adapting the extracted features as per the user's requirements and further adjusting the hyperparameters of the RBM to minimize processing and error in encoded values.

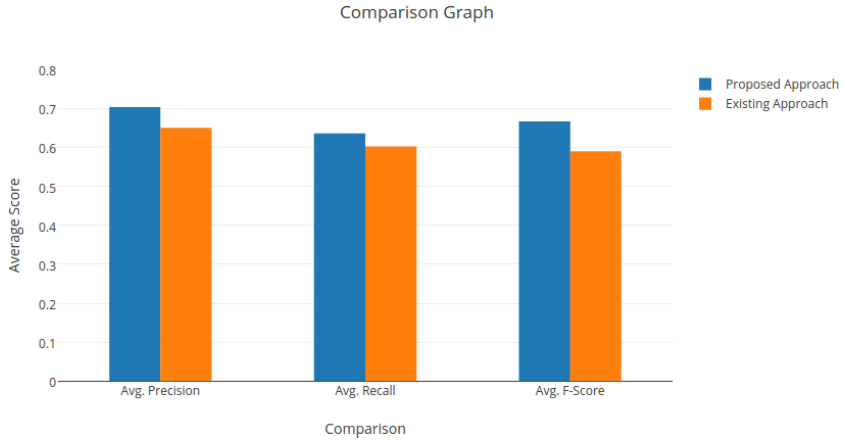


Fig. 6. Precison, Recall and F-Measure values for the proposed approach (*left bars*) and the existing approach (*right bars*).

Acknowledgments. We would like to extend our gratitude to Dr. Daya Gupta, Professor, Department of Computer Science and Engineering, Delhi Technological University(Formerly Delhi College of Engineering) for providing insight and expertise that greatly assisted this research.

References

1. Baxendale, P. B.: Machine-made index for technical literature—an experiment. IBM Journal of Research and Development, vol. 2, no. 4, pp. 354–361 (1958) doi: 10.1147/rd.24.0354
2. Berger, A., Mittal, V. O.: Query-relevant summarization using FAQs. In: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, pp. 294–301 (2000) doi: 10.3115/1075218.1075256
3. Chen, E., Yang, X., Zha, H., Zhang, R., Zhang, W.: Learning object classes from image thumbnails through deep neural networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 829–832 (2008) doi: 10.1109/ICASSP.2008.4517738
4. Chuang, W. T., Yang, J.: Extracting sentence segments for text summarization: A machine learning approach. In: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 152–159 (2000) doi: 10.1145/345508.345566
5. Darling, W. M., Song, F.: Probabilistic document modeling for syntax removal in text summarization. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 642–647 (2011)
6. Edmundson, H. P.: New methods in automatic extracting. Journal of the ACM, vol. 16, no. 2, pp. 264–285 (1969) doi: 10.1145/321510.321519
7. Hovy, E., Lin, C. Y.: Automated text summarization and the SUMMARIST system. In: Proceedings of a workshop on held at Baltimore. Association for Computational Linguistics, pp. 197–214 (1998) doi: 10.3115/1119089.1119121
8. Jin, F., Huang, M., Zhu, X.: A comparative study on ranking and selection strategies for multi-document summarization. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, pp. 525–533 (2010)

9. Learning, D.: Deep learning tutorial (2024) http://deeplearning.net/tutorial/
10. Luhn, H. P.: The automatic creation of literature abstracts. IBM Journal of research and development, vol. 2, no. 2, pp. 159–165 (1958) doi: 10.1147/rd.22.0159
11. Mani, I., House, D., Klein, G., Hirschman, L., Firmin, T., Sundheim, B.: The TIPSTER SUMMAC text summarization evaluation. In: Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics, pp. 77–85 (1999)
12. NLTK: Documentation. natural language toolkit for python (2023) http://www.nltk.org/
13. PadmaPriya, G., Duraiswamy, K.: An approach for text summarization using deep learning algorithm. Journal of Computer Science, vol. 10, no. 1, pp. 1–9 (2014) doi: 10.3844/jcssp.2014.1.9
14. Pandu, N., Prabhakar, R.: Performance evaluation of information retrieval systems (2024) web.stanford.edu/class/cs276/handouts/lecture8-evaluation.ppt
15. Shen, D., Sun, J. T., Li, H., Yang, Q., Chen, Z.: Document summarization using conditional random fields. In: Proceedings of the 20th International Joint Conference on Artifical Intelligence, vol. 7, pp. 2862–2867 (2007)
16. Singh, S. P., Kumar, A., Mangal, A., Singhal, S.: Bilingual automatic text summarization using unsupervised deep learning. In: International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), pp. 1195–1200 (2016) doi: 10.1109/ICEEOT.2016.7754874
17. Wan, X., Xiao, J.: Single document keyphrase extraction using neighborhood knowledge. In: Proceedings of the 23rd national conference on Artificial intelligence, vol. 2, pp. 855–860 (2008)
18. Wong, K. F., Wu, M., Li, W.: Extractive summarization using supervised and semi-supervised learning. In: Proceedings of the 22nd International Conference on Computational Linguistics, vol. 1, pp. 985–992 (2008)
19. Zhang, Y., Wang, D., Li, T.: iDVS: An interactive multi-document visual summarization system. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 569–584 (2011) doi: 10.1007/978-3-642-23808-6_37

ISSN 1870-4069

Classifications and Grammars of Simple Arabic Lexical Invariants in Anticipation of an Automatic Processing of this Language "The Temporal Invariants"

Dhaou Ghoul[1], André Jaccarini[2], Amr Helmy Ibrahim[1]

[1] Sorbonne University,
STIH Laboratory,
France

[2] Aix-en-Provence University,
MMSH,
France

{dhaou.ghoul, amrhelmyibrahim, andre.jaccarini}@gmail.com

Abstract. Our study focuses on the classification and treatment of simple Arabic lexical invariants that express a temporal aspect. Our aim is to create a diagram of grammar (finite state machine) for each invariant. In this work, we limited our treatment to only 13 lexical invariants. Our hypothesis begins with the principle that the lexical invariants are located at the same structural level (formal) as the schemes in the language quotient (skeleton) of the Arabic language. They hide a great deal of information and involve syntactic expectations that makes it possible to predict the structure of the sentence. To do this: First, for each lexical invariant, we developed a sample corpus that contains the different contexts of the lexical invariant in question. Second, from this corpus, we identified the different linguistic criteria of the lexical invariant that allow us to correctly identify it. Finally, we codified this information in the form of linguistic rules in order to model it by a diagram of grammar (finite state machine).

Keywords. Corpus, classification, syntactic environment, regular expression, Arabic language, lexical invariants, identification, linguistic rules, regular grammar, diagrams of grammars, NLP.

1 Introduction

As part of Mogador[1] project [1] we are interested in tool words that we call lexical invariants. Indeed, morphosyntactic ambiguity is most often identified in the Arabic language. According to our research, we noticed that several arguments demonstrate

[1] Modélisation des grammaires arabes, des données et des outils de recherche. https://halshs.archives-ouvertes.fr/halshs-00912009/document.

the ambiguity of tokens: For example, various graphical shape due to the absence of vowels and treating the word out of context. By contrast, a mechanism for morphosyntactic disambiguation of Arabic is needed to resolve the ambiguity of unveiled words. The aim of our research is to treat the Arabic language with minimum although complex rules without referring to a lexicon. In their work Audebert *et al.* proposed grammars that define some syntactic operators in Arabic (’inna, ’anna, ’an…) [2]. Our work is based on this bibliography to improve the grammatical basis of the operators.

In general, lexical invariants are classified according to their meaning and their function in the sentence. Consequently, these invariants play an important role in the interpretation of the sentence and the coherence of a text.

Our linguistic study consists in treating the context of these lexical invariants based on a corpus that represent a sample of each lexical invariant.

This study is organized as follows: In section 2: we present the concept of “lexical invariant” by exposing the various levels of invariance. Then, we classify the simple lexical invariants according to several criteria. In section 3: we present our method of linguistic study, which includes modeling through diagrams of the grammar’s some temporal lexical invariants. In section 4: we present two examples of simple lexical invariants. Conclusions and perspectives will be presented in section 5.

2 Definition and Classification of Simple Arabic Lexical Invariants

2.1 Definition of Lexical Invariant

As part of our research project, we try to project the Arabic language on its skeleton noted L/RAC. This skeleton is a quotient language whose elements are constituted of:

– Classes of equivalence of lexemes that are the schemes (patterns, lexical paradigm, wazn).
– Singletons (classes with only one element) which we can not associate a schemes (wazn). So, these singletons represent the invariants of the Arabic language that retain their same forms in the quotient language. This is why we have called "lexical invariants".

This language is an abstract object -*formally*- constructed from the free monoid constituted by the set of concatenations of the Arabic graphemes A* (with, A: Arabic alphabet and * designating the Kleene star) which the language L_{AR} consisting of the set of licit. Arabic graphic forms (graphic words separated by two whites) is a strict subset. What we mean by LEX is a particular subset of $L_{AR} \subset A^*$. Thus, $LEX \subset L_{AR} \subset A^*$. As well as, LEX is obtained by systematic segmentation of the elements of L_{AR}. [3]. Indeed, our appellation of “lexical invariant” is inspired by the definition of André Jaccarini [4].

Table 1. Temporal lexical invariants distribution.

	#sentences	**#words**	**#lexical invariant**	**#temporal invariant**	**%temporal invariant**
Text1	142	3460	1174	173	14.3
Text2	63	1311	507	80	15.7
Text3	123	2366	752	109	14.49
Text4	11	344	108	10	9.25

Let SC be the morphism associated with the operation of projection on the scheme.

We call the lexical invariant every element x belongs to LEX that is invariant with respect to the morphism SC.

The set of lexical invariants is denoted by:

$$INL=LEX \text{ inter } \{x ; SC(x)=\{x\}\}\{x ; x \in LEX \text{ et } SC(x)=\{x\}\}. \quad (1)$$

For example, SC (فإنهم/ *fa'inahum*) = ف SC (إن) هم = هم. إن. ف . With إن ϵ LEX, So إن is a lexical invariant.

These lexical invariants play a crucial role because they are the only elements that are both in the terminal vocabulary of L and L/RAC. They are on the same level as the schemes in L/RAC (quotient language) and they escape from the rules of morphological derivation. With, L is the language which consists of the set of grammatical sentences - not necessarily endowed with meaning-.

In Arabic there are two types of lexical invariants: simple lexical invariants like "*ḥattā*" and complex lexical invariants like "*ḥīnamā*". The latter are on the form "x *mā*" (with x representing a simple lexical invariant). In this study we treat only the simple lexical invariants.

2.2 Classification of Simple Temporal Arabic Lexical Invariants

Temporal lexical invariant classification remains an indispensable step in the automatic processing of the Arabic language. Based on literature on the one hand and on our study of these invariants on the other hand, we have tried to classify these 13 simple temporal invariants. Moreover, we calculated the percentage of temporal invariant's occurrences with respect to the total number of lexical invariants in four texts selected via our working corpus, which we created in previous work *[5, 6]*. Our calculation is made with the application "*Kawâkib*" *[7]* followed by a manual check.

The following table shows the temporal lexical invariants distribution with respect to all invariants in a given text.

The aim of this classification is to show the linguistic criteria of these types of words in Arabic and their influence in the construction of a text. The list of temporal invariants that we treat in this work is as follows: *l.m.ā; 'aṯnā'a; 'iḏā; 'iḏ; ḥattā; ḥīn/ ḥīna; lam; lan; munḏu; qad; ṯumma; baʿda; sawfa.*

Our classification of simple temporal lexical invariants is based on five linguistic criteria: agglutination, based on the word that follows them, syntactic role, ambiguity and based on their *rection* in the sentence.

Table 2. Lexical invariants classification based on their agglutination.

Class	Classification criteria	Example
1	Agglutination only to a coordinating conjunction.	*l.m.ā, ʾiḏ, ḥattā*
2	Agglutination to a coordinating conjunction and / or an interrogative conjunction.	*ʾiḏā, lam, lan*
3	Agglutination to a coordinating conjunction and / or a corroborating conjunction.	*qad*
4	Agglutination to a coordinating conjunctions and / or preposition and a personal pronoun attached.	*ʾaṯnāʾa*
5	Agglutination to a coordinating conjunctions and / or preposition and a personal pronoun attached and to a relative pronoun.	*ḥīn*
6	Agglutination to a coordinating conjunction and / or a personal pronoun attached and a relative pronoun.	*baʿda*
7	Agglutination to a coordinating conjunction and / or an interrogative conjunction and a corroborating conjunction.	*Sawfa*

Table 3. Lexical invariants classification based on the word that follows them.

Class	Classification criteria	Example
1	Pro-nominal invariants	*ʾaṯnāʾa, baʿda*
2	Pre-verbal invariants	*qad, lam, lan, sawfa*
3	Mixed invariants	*l.m.ā, ḥattā, ʾiḏ, ʾiḏā, ṯumma, munḏu, ḥīna*

Lexical invariants classification based on their agglutination with the prefixes or suffixes, we classified these 13 simple lexical invariants into 7 classes as the following table 2 shows.

Lexical invariants classification based on the word that follows them. According to our study of lexical invariants, it seemed useful to classify an invariant based on the word that follows them. Therefore, based on this criterion we have classified these 13 simple lexical invariants into three classes or families: pro-nominal invariants[2], pre-verbal invariants[3], mixed invariants[4] as the following table 3 shows.

Lexical invariants classification based on their syntactic roles. In Arabic language, we can find lexical invariants whose graphic form is unique but can have several syntactic functions. Among these lexical invariants is "*ḥattā*" whose unique graphic form may contain a subordinating conjunction, coordinating conjunction, adverb or preposition. Based on their syntactic roles, these lexical invariants are divided into 9 classes as the following table 4 shows.

Lexical invariant classification based on their "*rection*" in the sentence. Some lexical invariants in Arabic change the vocalization of the word that follows them. This

[2] The lexical invariants that precede the noun or the elements equivalent to the noun.

[3] The lexical invariants that precede the verb or the elements that can receive the verb.

[4] The lexical invariants that precede either a verb or a name or another invariant.

Table 4. Lexical invariants classification based on their syntactic roles.

Class	Syntactic class	Example
1	Subordinating conjunction	*lammā, lamā, limā, ’iḏ, ḥattā, ’iḏā, ’aṯnā’a, ḥīna, munḏu*
2	Coordinating conjunction	*ḥattā, ṯumma*
3	Negation particle	*lan, lam, lammā*
4	Interrogative pronoun	*limā*
5	Preposition	*ḥīna, ’aṯnā’a, baʿd, ḥattā, munḏu*
6	Temporal adverb	*l.m.ā, ’iḏ, ’iḏā, ’aṯnā’a, ḥīna, munḏu, baʿd, sawfa, ḥattā, lan, lam, ṯumma, qad*
7	Conditional conjunction	*’iḏ, ’iḏā*
8	Surprise particle	*’iḏ, ’iḏā*
9	Future particle	*sawfa*

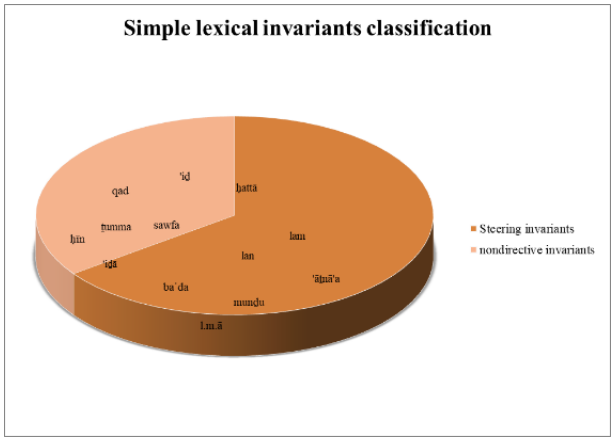


Fig. 1. Lexical invariants classification based on their “reaction” in the sentence.

linguistic criterion is useful in the identification of these lexical invariants. It is therefore important to classify these invariants according to their power of reaction. Indeed, and based on this criterion, we classified these lexical invariants into two classes as the following figure (Fig.1) shows:

Lexical invariant classification based on their ambiguity. The vowels absence in the Arabic text is one of the major problems of automatic processing of this language. This problem leads to several cases of grammatical (syntactic) and graphical or even semantic ambiguity. Moreover, the phenomenon of agglutination in Arabic can be a major source of ambiguity. Therefore, it seemed useful to classify our study invariants according to their ambiguity (graphic, syntactic and semantic) as the following table 5 shows.

It is worth noting that the different abbreviations are illustrated in the appendix. In general, the lexical invariants classification depends on their linguistic properties. Therefore, this classification is always debatable and not exhaustive.

Table 5. Lexical invariants classification based on their ambiguity.

Lexical invariant	Graphic ambiguity	Syntactic ambiguity	Semantic ambiguity
l.m.ā	*lammā, limā, lamā*	CS, PNE, ADVT	Correspondence, temporal, negation
’iḏ	*’iḏ, āḏ*	CS, COND, ADVT, PS	Condition, surprise
’iḏā	*’iḏā, āḏā, āḏan, ’iḏan*	CS, PS, CCOND, ADVT	Condition surprise
lan	Unique	ADVT, PNE	Negation
lam	Unique	ADVT, PNE	Negation
ḥattā	Unique	CS, CC, PREP, ADVT	Coordination, intensity, justification, direction…
ṯumma	*ṯumma, ṯamma*	CC, ADVT, PD, Verb	Ranking (order)
munḏu	Unique	CS, ADVT, PREP	Time report, temporal interrogation
qad	Unique	ADVT	Certainty, uncertainty
ḥīna	ḥīna, ḥīn	ADVT, CS, PREP	Temporal correspondence, a moment
’aṯnā’a	*‘aṯnā’a, aṯnā’*	CS, ADVT, PREP	Temporal synchronization
	baʿda,	PREP	Temporal, Continuity
bʿd	baʿdu,	ADVT	Anteriority
	buʿd,	Noun	Distance
	baʿida, baʿuda, baʿada	Verb	Move away
sawfa	Unique	ADVT, PF	Promise

3 Methodology of Lexical Invariants Modeling by Grammars Diagram

We will start from the idea that a grammar is a point of view among others on the language and that these points of view are related to the tasks assigned to these grammars. The grammars of the invariants are made up on them and their linguistic analysis. Information about them is largely known by Arab and Western grammarians. However, our contribution is summarized in the angle of analysis that we adopt and the examples that we select.

So, the grammars presented in our work are intended for the automated morpho-syntactic analysis of the Arabic language. The idea here is to transform the linguistic rules of each invariant that we have defined previously into a non-fixed-form formal grammar. These grammars describe the syntactic expectations of invariants in Arabic [8].

3.1 Different Levels of Our Method

Our method of Arabic lexical invariant modeling is located as part of the theory of “abstract machines”. It is based on two major phases (two levels of analysis).

The linguistic level. Our methodology of the study linguistic of each lexical invariant consists to study its environment based on a sample of representative sentences. The different steps of our method are:

- Identification of lexical invariants in the corpus.
- Treatment of the environment (left and right context) of the lexical invariant.
- Develop hypotheses on the structure of the sentence based on the principle of the "empty sentence".
- Identify the different characteristics of the lexical invariants (grammatical role, ambiguity, suppressibility or not, graphical form, distributional analysis [9, 10], local and global scope...).
- Study of the syntactic expectations of lexical invariants.
- Codify the different information into non-fixed formal grammars (rules of production or linguistic rewriting), to bring out the relations in order to make them usable by the machine.

Note that the semantic part of each invariant will be studied in parallel and can also be related to the definition of discriminating criteria.

The graphic level. This level consists in modeling the grammars in the form of finite state automata (grammars schemes) in order to make it comprehensible by a machine. The graphs represent successions of lexical invariant's syntactic expectations. This means that one passes from a state represented by a node to another state by means of a transition which is symbolized by an arc. These arcs are labeled with the terminal symbols of the grammar.

So why use the theory of automata to represent or treat the Arabic language?

Indeed, the structural (strongly grammatical) characteristics of Arabic are also algorithmic characteristics. These characteristics are easily translatable as part of the theory of automata. Thus, the theory of automata offers the possibility of a good study of Arabic in the universe of knowledge.

3.2 Limits of Our Method

In a general way, our method, purely algorithmic, differs from those of contemporary research. It uses minimal resources and is independent of lexicons. The origin of this approach is due to the representation of the Arabic language in the form of a quotient or skeleton language. The latter can be obtained by reducing all the roots into a single control root "faʿala" to represent all Arabic schemes.

Indeed, at this stage we have seen that in the language quotient exist particular words that we note by "lexical invariants". Our method consists in a surface analysis of these invariants in order to construct non-fixed formal grammars that represent the environment of each invariant. These grammars can be linked and combined. However, our method presents some limits:

- "Ad hoc" construction designed for a given task: The automata may contain unnecessary parts.

- Difficulty in marking the meaning of the lexical invariant in the automaton.
- Grammar diagrams may in some cases produce grammatical correct, but semantically incorrect sentences (invalid sentences because of the presence of a succession of epsilons).

4 Examples of Simple Temporal Lexical Invariants

In this section we will present two examples of simple lexical invariants among the 13 invariants that we have analyzed in this work. First of all, we will extract the maximum of information in the form of linguistic rules. Then, we will model these rules in regular grammars acceptable by finite state automata.

4.1 Lexical Invariant "*ḥīn /ḥīna*"

Outside of its context and in the absence of vowels, the word "*ḥīn*" is ambiguous grammatically. Indeed, this word may be a temporal subordinating conjunction or a noun. The disambiguation of this word depends on its position in the sentence as well as on the linguistic study of its environment. To assign the correct grammatical function to "*ḥīn*", we must treat its context based on a sample corpus. This sample corpus represents an extract of 60 sentences from our working corpus [5, 6].

Independently of its grammatical function, the removal of "ḥīn" causes the disruption of the structure as well as the meaning of the sentence.

Temporal subordinating conjunction. In this case, the lexical invariant "*ḥīn*" expresses a relation of time between two events and noted "*ḥīna*". This is a link between two parallel events as shown in the following example (1). In addition, it is very fertile lexical invariant. That is, it can agglutinate at n + 1 with suffixes like "*mā*," "*iḏan*", "*ḏāk*" to establish a new lexical invariant.

أصابته الدهشة حين زار مدينة القاهرة » *'asābathu aldahšat hīna zāra madīnat alqāhirat*.» (He was astonished when he visited the city of Cairo). (1)

أصابته الدهشة عندما زار مدينة القاهرة » *'asābathu aldahšat ʿindamā zāra madīnat alqāhirat*. » (He was astonished when he visited the city of Cairo). (1')

From the examples (1) and (1') above, we note that we can replace "*ḥīna*" by "*ʿindamā*" without any change in the sentence either in the sense or in the structure. However, this permutation is only valid in the case where "*ḥīna*" has the same syntactic expectations (a verb phrase) as "*ʿindamā* "(sentence 1'). This permutation is strictly forbidden in the case where "*ḥīna*" is followed by a nominal group as shown in the example (2):

في حين أن الكثير من الشباب ملتزمون بجلسات المقاهي » *fī ḥīna 'anna alkaṯīr min alšabāb multazimūn bigalasāt almaqāhī*.» (However, many young people were keen on going to coffees). (2)

If we look at the example (2), the sequence "*fī ḥīna*" is equivalent to "*baynamā*" (2') which marks a temporal opposition between two non-identical events. So, the lexical invariant "*ḥīna*" does not keep the same meaning when it appears in a sequence of invariants like "*fī ḥīna*". It no longer expresses temporality, but it expresses an opposition.

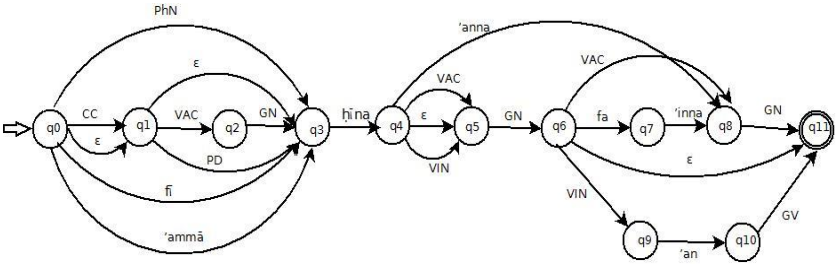


Fig. 2. Grammar diagram of "*ḥīna*" temporal subordinating conjunction.

بينما أن الكثير من الشباب ملتزمون بجلسات المقاهي « *baynamā 'anna alkaṯīr min alšabāb multazimūn bigalasāt almaqāhī*.» (By contrast, many young people regularly attended cafés). (2')

As for syntactic expectations, the lexical invariant "ḥīna" in this case, waits in n +1 either a verb phrase (the verb can be either in the present tense or in the past tense), or a nominal group. However, it can be preceded in n-1 by a coordinating conjunction, the preposition "*fī*", a demonstrative pronoun, corroborating conjunction "*ammā*" or the preposition "*li*". Note that in general if this invariant preceded in n-1 by the invariant "*fī*", it will be followed in n + 1 by "'*anna*".

The Formal grammar that accepts phrases in which "*ḥīna*" plays the role of a temporal subordinating conjunction is presented in the form of a finite automaton comprising 12 states as the following figure (Fig.2) shows. G (*ḥīna*, CST)= {N, ∑, P, q0, q11}, with N= {q0, q1, q2, q3, q4, q5, q6, q7, q8, q9,q10, q11} is the not terminal vocabulary, ∑= {VAC, VIN, GN, GV, CC, PD, PhN, *ḥīna*, *'inna*, *'anna*, *'an*, *'ammā*, *fī*, *fa*, ε} is the input vocabulary, P is the set of production rules and q0ϵN is an initial state, q11ϵN is the final state.

G (*ḥīna*, CST) =

1. q0→ CC q1| PhN q3|fī q3|'ammā q3|ε q3,
2. q1→ VAC q2|PD q3|ε q3,
3. q2→ GN q3,
4. q3→ ḥīna q4,
5. q4→ VAC q5| VIN q5|'anna q8|ε 5,
6. q5→ GN q6,
7. q6→ VAC q8|VIN q9|fa q7|ε q11,
8. q7→ 'inna q8,
9. q8→ GN q11,
10. q9→ 'an q10,
11. q10→ GV q11.

Noun (ḥīn). In general, the lexical invariant "*ḥīn*" occupies the function of a name in the case where it intervenes either in the middle or at the end of the sentence as shown in the following examples (3) and (4):

ويراك في كل حين وكل لحظة « *wa yarāka fī kulli ḥīn wa kulli laḥẓat*.» (He always saw you at all times). (3)

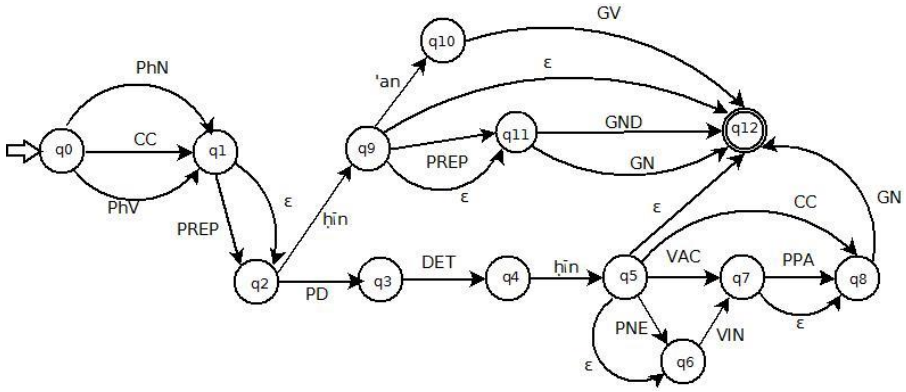


Fig. 3. Grammar diagram of "*ḥīn*" noun.

ولتعلمن نبأه بعد حين « *wa litaʿlamanna nabaʾahu baʿda ḥīn*. » (You will soon hear about it). (4)

According to the two sentences above (3 and 4), we note that "*ḥīn*" expresses the notion of a short or long moment. That is why we can be replaced it by the word "*waqt*" (moment). So in this case, "*ḥīn*" has a syntactic and semantic equivalence with the word "*waqt*" as shown in the examples (3') and (4').

ويراك في كل وقت وكل لحظة « *wa yarāka fī kulli waqt wa kulli laḥẓat*.» (He always saw you at all times). (3')

ولتعلمن نبأه بعد وقت « *wa litaʿlamanna nabaʾahu baʿda waqt*. » (You will soon hear about it). (4')

Note that in this case "*ḥīn*" can concatenate with the article "*Al*" to transform it from a noun to a determined noun as shown in the following sentence (5) or with the preposition "*li*" (example 6).

فحتى ذلك الحين أترككم في رعاية الله « *faḥattā ḏalika Alḥīn ʾatrukukum fī riʿāyat allah*. »(And in the meantime, may God protect you). (5)

ينتظرون لحين ان يأتي دورهم في الحكاية « *yantaẓirūn liḥīn ʾan yaʾtī dawruhum fī Alḥikāyat*. » (They were waiting for a moment to come to their role in history). (6)

Similarly, in this sentence (5) if the invariant "ḥīn" is replaced by the word "waqt", the structure and meaning of the sentence do not move (Example 5'). Consequently, we can permute the sequence "liḥīn" with the lexical invariant "ilā" or "ḥattā" without any change in the source sentence (example 6').

فحتى ذلك الوقت أترككم في رعاية الله « *faḥattā ḏalika Alwaqt ʾatrukukum fī riʿāyat allah*. » (And until that time God bless you). (5')

ينتظرون إلى ان يأتي دورهم في الحكاية « *yantaẓirūn ʿilā ʾan yaʾtī dawruhum fī Alḥikāyat*. » (They will wait for the time that will come their role in the history). (6')

In this case, the syntactic expectations of "*ḥīn*" are less complex than those of a subordinating conjunction. Indeed, it expects in n + 1 either the empty set or a nominal group. This nominal group can be in the form "*min* GND" (example 7).

هل اتى على الانسان حين من الدهر « *hal ʾatā ʿalā Alʾinsān ḥīn min Aldahr*. » (Did the man go through a time lapse). (7)

The Formal grammar that accepts phrases in which "*ḥīna*" plays the role of a temporal subordinating conjunction is presented in the form of a finite automaton

Table 6. Linguistic Properties of "*ḥīn*".

ḥīn /ḥīna		
Graphic form / grammatical class	ḥīna /CST	ḥīn / Noun
Meaning	Temporal: *fī alwaqt allaḏī*	Expresses the notion of a moment
Removal	No	No
Position in the sentence	Beginning or middle	Middle or end
Change of position	Yes	No
Agglutination in n-1	*wa, fa*	*li, Al*
Agglutination in n+1	*mā, ḏāk, ’iḏan*	No
Form of sentence	*ḥīn* P Q ou Q *ḥīn* P	P *ḥīn* ou P *ḥīn* Q
Switching with other invariants	*ʿindamā* *fi ḥīna ’anna =baynamā* *liḥīn= ḥattā*	*waqt*[5] *liḥīn= ʿilā*
Syntactic expectations at n + 1	GV or GN	GN, min GND or Ø
Governance	Governs two kernels: verb + subject and / or theme + predicate.	No
Discontinuous	No	No
Traduction	When	Moment

comprising 13 states as the following figure (Fig.3) shows. G (*ḥīn*, Noun)= {N, ∑, P, q0, q12}, with, N= {q0, q1, q2, q3, q4, q5, q6, q7, q8, q9, q10, q11, q12} is the not terminal vocabulary, ∑= {VAC, VIN, GN, GND, GV, CC, PD, DET, PREP, PNE, PPA, PhN, PhV, *ḥīn*, *’an*, ε} is the input vocabulary, P is the set of production rules and q0ϵN is an initial state, q12ϵN is the final state.

G (*ḥīn*, Noun) =

1) q0→PhN q1| PhV q1| CC q1,
2) q1→ PREP q2|ε q2,
3) q2→PD q3|ḥīn q9,
4) q3→DET q4,
5) q4→ ḥīn q5,
6) q5→ VAC q7|PNE q6|CC q8|ε q6|ε q12,
7) q6→ VIN q7,
8) q7→ PPA q8|ε q8,
9) q8→GN q12,
10) q9→ PREP q11| ’an q10|ε q11|ε q12,
11) q11→ GV q12.

The different linguistic properties of "*ḥīn*/ *ḥīna*" are summarized in the table below:

[55] In this case, the word "waqt" does not designate a lexical invariant but rather a noun.

Table 7. Linguistic Properties of "*ʾaṯnāʾa*".

ʾaṯnāʾa	
Graphic ambiguity	Unique :أثناء «*ʾaṯnāʾa*», اثناء «*aṯnāʾa*»
Grammatical ambiguity	Unique: Preposition
Meaning	Temporal: *ḫilāla*
Removal	No
Position in the sentence	Beginning or middle
Change of position	Yes
Agglutination in n-1	*wa, fa, ka*
Agglutination in n+1	Attached personal pronoun: *h, hā…*
Form of sentence	*ʾaṯnāʾa* P Q ou Q *ʾaṯnāʾa* P
Switching with other invariants	Yes: *ḫilāla*
Syntactic expectations at n + 1	GND
Governance	No
Discontinuous	No
Traduction	During

4.2 The Lexical Invariant "*ʾaṯnāʾa*"

According to Arabic literature, the word "*ʾaṯnāʾa*" is a preposition that connects two or more events in the sense of "*ḫilāla*". These events are parallel and simultaneous with respect to time. If we take into account how to transcribe the "Hamza", the lexical invariant " *ʾaṯnāʾa* " is not graphically ambiguous.

The aim of our study is to have more or less precise idea on this lexical invariant. To do this, we first selected some samples of sentences that contain " *ʾaṯnāʾa* " from our corpus [5, 6]. Our study sample of " *ʾaṯnāʾa* " contains 50 sentences. We used this sample at first to identify the notable features. Among these pertinent remarks, we found that " *ʾaṯnāʾa* " was always followed by a determined nominal group.

Indeed, to better understand the functioning of this invariant in the Arabic language, we have treated some selected examples from our study sample.

Consider the following examples:

وأثناء دراستي في بريطانيا قدمنا عملا كنت أعمل فيه مساعد مخرج » *wa ʾaṯnāʾa dirāsatī fī brīṭānyā qaddamnā ʿamalāa kuntu ʾaʿmal fīh musāʿid muḫrig* ». (During my studies in the United Kingdom, we presented a work in which I was an assistant director) (7).

يأخذ الرواية معه أثناء سفره إلي الخارج » *yaʾḫuḏu alriwāyat maʿahu ʾaṯnāʾa safarihi īllā alḫārig*» (He took the novel with him during his travels abroad). (8)

From examples above, we note that " *ʾaṯnāʾa* " can intervene either at the beginning or in the middle of the sentence and never at the end of the sentence except in the case where it is agglutinated in n + 1 with a Personal pronoun attached as shown in the following sentence:

فليحرص المسلم النبيه النزيه على محاسبة نفسه المحاسبة الشاملة قبل العمل وأثناءه» *faliyaḥuṣ almuslim alnabīh alnazīh ʿalā muḥāsabat nafsihi almuḥāsabat alšāmilat qabla alʿamal wa ʾaṯnāʾahu* » (The true Muslim must be vigilant in asking for complete accounts before and during the work). (9)

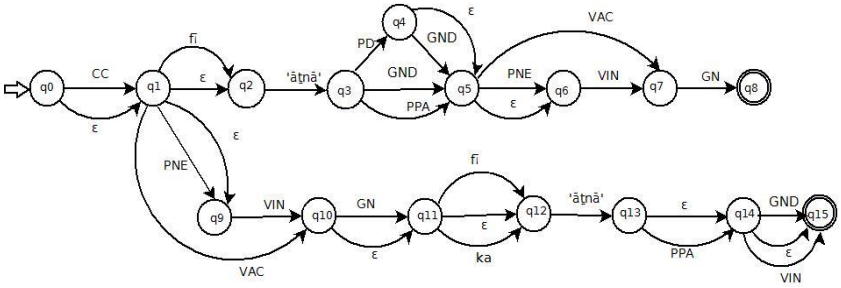


Fig. 4. Grammar diagram of "*'aṯnā'a*" *subordinating conjunction.*

As we noted above, this invariant takes the meaning of "*ḫilāla*". That is, we can switch "*'aṯnā'a"* by *"ḫilāla"* and keeping the same meaning and structure of the sentence as shown in the following example:

يأخذ الرواية معه **أثناء** سفره إلي الخارج ↔ يأخذ الرواية معه **خلال** سفره إلي الخارج (He took the novel with him during his travels abroad).

The table below summarizes these different linguistics properties:

The Formal grammar that accepts phrases in which "*'aṯnā'a* " plays the role of a temporal subordinating conjunction is presented in the form of a finite automaton comprising 16 states as the following figure (Fig.4) shows. G (*'aṯnā'a*, CS)= {N, ∑, P, q0, q15}, with N= {q0, q1, q2, q3, q4, q5, q6, q7, q8, q9, q10, q11, q12, q13, q14, q15} is the not terminal vocabulary, ∑= {VAC, VIN, GN, CC,PPA, PNE, PD, fī, ka, 'aṯnā'a, ε} is the input vocabulary, P is the set of production rules and q0ϵN is an initial state, q15ϵN is the final state.

G (*'aṯnā'a*, CS) =

1) q0→ CC q1| ε q1,
2) q1→ fī q2 | VAC q10| PNE q9|ε q2|ε q9,
3) q2→ 'aṯnā' q3,
4) q3→ PD q4|GND q5| PPA q5,
5) q4→ GND q5|ε q5,
6) q5→ PNE q6| VAC q7|ε q6,
7) q6→ VIN q7,
8) q7→ GN q8,
9) q9→ VIN q10,
10) q10→ GN q11|ε q11,
11) q11→ fī q12| ka q12|ε q12,
12) q12→ 'aṯnā' q13,
13) q13→ PPA q14|ε q14,
14) q14→ GND q15| VIN q15| ε q15.

5 Conclusions and Perspectives

In this work, we tried to study and classify 13 simple Arabic lexical invariants. In our study, we have focused on the notions of "permutation" or "commutation" that based the approach of structural linguistics called "distributional". To do this, first for each lexical invariant, we developed a sample corpus that contains different contexts of the

lexical invariant in question. Second, from this corpus, we identified the different linguistic criteria of the lexical invariant that allow us to correctly identify it.
Finally, we codified this information in the form of linguistic rules in order to model it by diagram of grammar (finite state machine). Indeed, as perspectives for this work, we will evaluate our temporal invariant grammars on a big data.

6 Appendix

VIN: Verb present tense, VAC: verb past tense, VI: imperative verb, N: Noun, PREP: preposition, CC: coordinating conjunction, CS: subordinating conjunction, ADVT: adverb of time, PNE: negation particle, CCOND: conditional conjunction, CST: temporal subordinating conjunction, PS: Particle of surprise, PD: demonstrative pronoun, PF: Future particle, PPA: Pronoun personal attached. GN: Nominal group, GND: Determined nominal group, GV: verbal group, PhN: Nominal sentence, PhV: verbal sentence, DET: determiner.

References

1. Jaccarini, A., Gaubert, C.: Le programme Mogador en linguistique formelle arabe et ses applications dans le domaine de la recherche et du filtrage sémantique (2012)
2. Audebert, C., Jaccarini, A.: À la recherche du khabar, outils en vue de l'établissement d'un programme d'enseignement assisté par ordinateur. Institut français d'archéologie orientale du Caire, pp. 217–256 (1986)
3. Ghoul, D.: Classifications et grammaires des invariants lexicaux arabes en prévision d'un traitement informatique de cette langue les invariants temporels. Doctorat (2016)
4. Jaccarini, A.: Grammaires modulaires del'arabe. Mise en œuvre informatique et stratégies. Thèse de doctorat, Sorbonne (1997)
5. Ghoul, D.: Construction d'un corpus arabe à partir du Web dans le but d'identifier les mots-outils ou tokens. JADT: Journées internationales d'Analyse statistique des Données Textuelles, INALCO, pp. 271–276 (2014)
6. Ghoul, D., Ibrahim, A. H.: Web arabic corpus : construction d'un large corpus arabe annoté morpho syntaxiquement à partir du Web. CECTAL'15, pp. 12–16 (2015)
7. Gaubert, C.: Kawâkib, une application pour le traitement automatique de textes arabes. no. 44, pp. 66–81 (2010)
8. Ghoul, D, Ibrahim, A. H., Audebert, C.: Rules-based grammatical and semantic disambiguation of the token "hatta". In: 5th International Conference on Information Communication Technology and Accessibility (ICTA), pp. 1–6 (2015)
9. Dubois, J., Dubois-Charlier, F.: Principes et méthode de l'analyse distributionnelle, Langages, vol. 5, no. 20, pp. 3–13 (1970)
10. Dubois, J.: Grammaire distributionnelle, vol. 1, no. 1, pp. 41–48 (1969)

ISSN 1870-4069

The Fix-point of Dependency Graph – a Case Study of Chinese-German Similarity

Tiansi Dong[1], Armin B. Cremers[1], Juanzi Li[2], Peiling Cui[3]

[1] University of Bonn,
B-IT,
Germany

[2] Tsinghua University,
Department of Computer Science,
P. R. China

[3] University of Bonn,
Department of Chinese,
Germany

dongt@bit.uni-bonn.de, abc@iai.uni-bonn.de,
lijuanzi@mail.tsinghua.edu.cn, pcui@uni-bonn.de

Abstract. Applications using linguistic dependency graphs (LDG) produce exciting results in neural-NLP and machine translation, as it captures more semantic information. A case study is carried out to make *language dependency graphs*(LDG) closer to meaning representations. We designed graph-transformation rules, which remove some syntactic covers, and keep updating an LDG, till a fix-point is reached, which we name *spatial linguistic graph* (SLG). The formal definition of SDG is presented. An evaluation using SimRank-based method is conducted using paired German-Chinese sentences in a grammar book (totally 682 paired sentences). Results show that SDGs of paired sentences are more similar than that of LDGs – supported by 89.4% observations. After removing 66 invalid inputs, the support reaches to 99.03%. Comparison with related work is presented. Applications of SDG in word-embedding and Machine Translation are described.

Keywords: Linguistic/spatial dependency graph, fixed-point, SimRank-similarity.

1 Introduction

Recent NLP applications using linguistic dependency graphs (LDG) produce quite exciting and positive results. For example, in neural network computing, the word-embedding based on dependency graph is more accurate than sequence model, i.e., [19]; graph-based learning improves the quality in statistical machine translation, [2]; dependency-based machine translation demonstrates significantly better results than the phrase-based model, i.e., [28, 20]. The reason lies the fact that LDG captures long-distance word relations, and reveals more explicitly semantic relations.

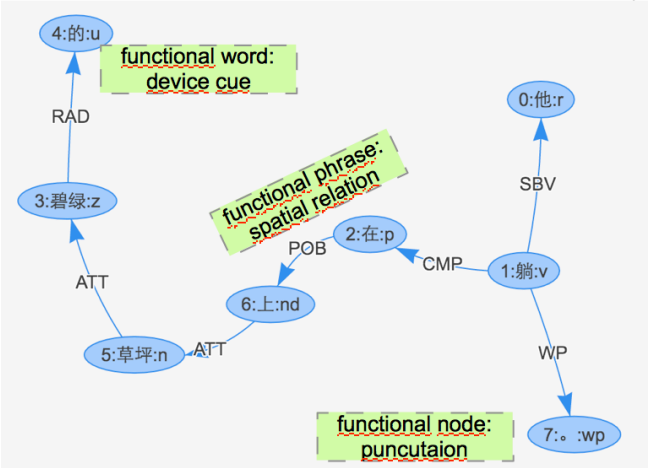
Fig. 1. A scene of a man lying on the grass.

The questions raised in this paper are: How close (or far away) is an LDG to the semantic representation? Can we make an LDG closer to its intended semantic representation?

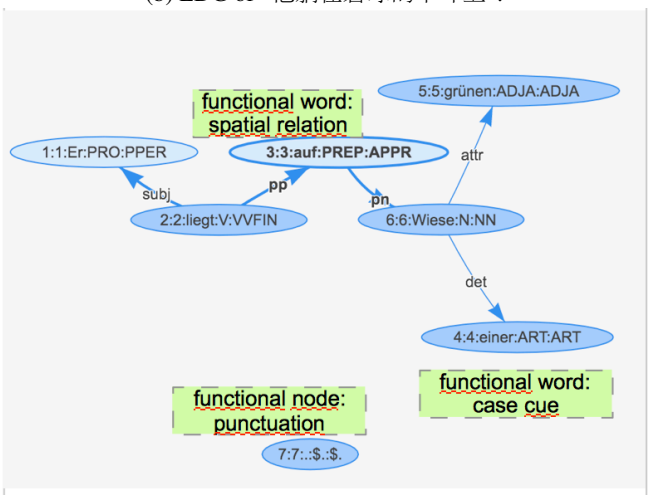
Let us start with a thought experiment is as follows: a bilingual speaker observes a scene, as illustrated in Fig 1, and describes the scene in two languages. He may give descriptions in two languages: For the Chinese description is "他/he 躺/lies在碧绿/green的草坪/lawn上。", its German description is "Er liegt auf einer grünen Wiese.". Underlined words are content words. How similar are the two dependency graphs? Can we make them more similar? We would assume that his descriptions have the same meaning. Their raw linguistic structures are shown in Figure 2 (a,b); the intended semantic structure shall be ideally very similar, as illustrated in Figure 3 (a, b).

The theoretical background can be found in the research of the relations between language and space. Language and thoughts are rooted in the knowledge of the space around us, [12]. Space structures linguistic descriptions, [31]; On the other hand, linguistic descriptions select and highlights some aspects in the space, i.e., [30]. Given a description about the same spatial layout, can we extract its spatial semantics? Research can be dated back to the work on relations between language form and meaning, in term of *cue*, i.e., [21]. Cross-linguistic studies showed that each language uses a particular set of cues: Italian extremely relies on agreement cues, German relies on both agreement cues and animacy cues, English relies overwhelmingly on word order, e.g., [22], Chinese relies on cues of word order, passive marker 被/by, animacy, object marker 把/hold, indefinite marker 一/one, i.e., [21].

The work reported in this paper is a case study in the setting of Chinese and German – two languages that use completely different cues. We collected manually all paired-sentences (682 pairs) in a Grammar book, [29], so that different linguistic structures are covered, and transformed them into dependency graphs using existing tools. Then, we manually designed sub-graph transformation rules by walking through all these sentences, based on different cues in the two languages. We also proposed SimRank-based similarity measurement to evaluate the results. The experiment system transforms the structures in Figure 2 (a, b) into Figure 3 (a, b), which we call *spatial dependency graph* (SDG).



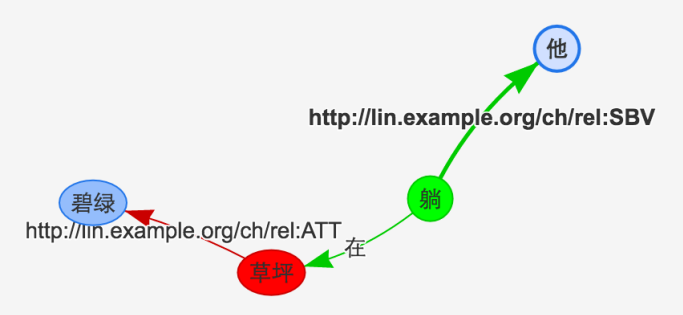
(b) LDG of “他躺在碧绿的草坪上”.



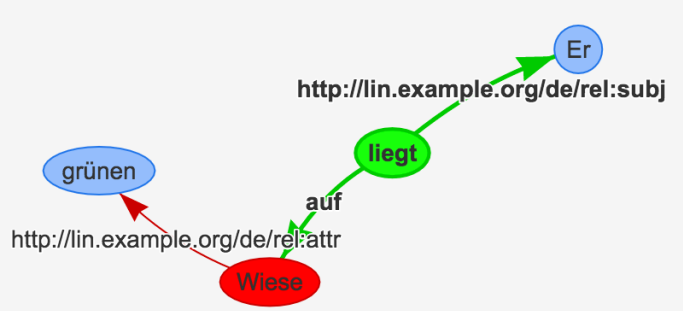
(b) LDG of “er liegt auf einer grünen Wiese”.

Fig. 2. The linguistic dependency graphs of a paired sentences. (a) is based on http://www.ltp-cloud.com/; (b) is based on http://kitt.cl.uzh.ch/kitt/parzu/.

The rest of the paper is structured as follows: section 2 defines the spatial dependency graph (SDG) as a fixed point of graph-transformations of LDG, and proposes the similarity conjuncture – if SDG is closer to semantic representation, SDGs of paired sentences shall be more similar than their LDGs; section 3 describes the SimRank-based method for graph comparison; section 4 show the experiment using paired sentences in a Chinese-German grammar book; section 5 listed some related work in the literature; section 6 presents two promising applications of SDGs: word-embedding, and machine translation.



(a) SDG of “他躺在碧绿的草坪上”.



(b) SDG of “er liegt auf einer grünen Wiese”.

Fig. 3. The spatial dependency graphs Fig 2 (a) and Fig 2 (b).

2 Spatial Dependency Graph and the Cross-linguistic Similarity Conjuncture

The starting point is language dependency graph (LDG) of a sentence, e.g., [18, 9]. Our work is to identify pure-syntactic structure (from a single node, to a sub-graph) in LDG, and repeatedly update them, till a fixed-point is reached. If it is a leaf node, we remove it, and set it as a feature of its parent node; if it bridges two nodes, we create a new edge from its parent node to its child node, and move it as one label of this node edge, as illustrated in Figure 2 and Figure 3. We formally define a LDG as follows.

Definition 1 A language dependency graph, LDG, is a *directed*, *labeled*, and *non-cyclic* graph $g = \langle V, E, \mu, \nu \rangle$, V is the set of nodes, E is the set of *directed*, *labeled*, and *non-cyclic* edges between nodes; $\mu : V \to \vec{F}$ is a function for node features – at least three feature: position, word form, and parts of speech; $\nu : E \xrightarrow{l} L$ is an edge-labelling function. POS is the set of parts of speech, POS_{cnt} is the set of parts of speech of content words, fun is the set of parts of speech of functional words, satisfying $POS = POS_{cnt} \cup POS_{fun}$ and $POS_{cnt} \cap POS_{fun} = \emptyset$. A directed edge from u to v, labeled with $l = \nu(u, v)$, non-cyclic ($u \neq v$) is written as $u \xrightarrow{l} v \in E$. Edges are *deterministic*. That is, for any u, v in V, there is only one edge from u to v labeled with l ($u \xrightarrow{l} v \in E$). Formally, $\forall u, v, v' \in V[u \xrightarrow{l} v \in E \wedge u \xrightarrow{l} v' \in E \Rightarrow v = v']$.

Our general observation on the data-set of deciding whether a node is a *cue* is to observe the parts of speech of the word in the node, and the number of children. If it has two or more than two children, it is not a *cue* node; if the node has a functional word, and one or less than one child node, it is a *cue* node, and shall be updated. An intuitive understanding is that a node with more than two children will served as a semantic frame on the relation among itself and the children. To identify whether a node is a *cue*, we need to consider *pattern* of, and around a node, as defined below.

Definition 2 Let $\vec{f}$ be a feature vector of node, at least one component has non-empty value, l be a non empty label. A pattern of $g = \langle V, E, \mu, \nu \rangle$ is recursively defined as follows: (1) $\mu(?v) = \vec{f}$ is a pattern, read as 'a node $?v$ has feature $\vec{f}$'; (2) $\nu(?u \rightarrow ?v) = l$ is a pattern, read as 'an edge from $?u$ to $?v$ having label l; (3) if c is a pattern, $\neg c$ is a pattern; (4) if c_1, c_2 are patterns, $c_1 \wedge c_2$, $c_1 \vee c_2$ are patterns; (5) patterns are only in these forms. $?v$, $?u$ are variables. Let c be a pattern, $var(c)$ is the set of all the variables in c, $node(c)$ is the set of all nodes in c.

After having identified a pattern in a graph, we need to bind variables in the pattern with the concrete nodes in the graph. Inversely, if we have a pattern, and a list of node assignment of its variable, we can construct a concrete graph from this pattern using these assignment.

Definition 3 Let c be a pattern, g be a graph has pattern c. The *binding* of pattern c with graph g is the assignment of each element in $var(c)$ by the corresponding node of g. A graph *construction*, based on pattern c, using bindings b, is to replace node variables in c with the bounded values defined in b.

A process of updating *cue* node is then a process of pattern identification within a graph, get binding to the graph, and construct a new sub-graph based on a new pattern.

Definition 4 A *transformation* $f_{c_1 \rightarrow c_2}$, satisfying $var(c_1) = var(c_2)$, is a function from graph to graph – to replace each sub-graph with c_1 pattern with a graph constructed by c_2 pattern using the binding of pattern c_1.

We strictly require that $var(c_1) = var(c_2)$, instead of $var(c_1) \supseteq var(c_2)$, so that inverse operation can be defined. For the transformation $f_{c_1 \rightarrow c_2}(g) = g'$, some nodes in g may be placed as label information in g', instead of being simply removed.

Definition 5 The *Identity transformation* f_{id} is defined as $f_{c \rightarrow c}$, for any c.

It is obvious that for any graph g, pattern c, $f_{c \rightarrow c}(g) = g$.

Definition 6 Let $f_{c_1 \rightarrow c_2}$ and $f_{c_3 \rightarrow c_4}$ be two transformations, its inverse transformation $f^{-1}_{c_1 \rightarrow c_2}$ is defined as $f_{c_2 \rightarrow c_1}$. The iteration, $f_{c_1 \rightarrow c_2} \circ f_{c_3 \rightarrow c_4}$, is defined as: for any g, $f_{c_1 \rightarrow c_2} \circ f_{c_3 \rightarrow c_4}(g) = f_{c_1 \rightarrow c_2}(f_{c_3 \rightarrow c_4}(g))$.

Theorem 1 Transformation is associative.

Proof 1 Let f_1, f_2, f_3 be three transformations, $(f_1 \circ f_2) \circ f_3(g) = (f_1 \circ f_2)(f_3(g)) = (f_1(f_2(f_3(g)))) = (f_1(f_2 \circ f_3(g))) = (f_1 \circ (f_2 \circ f_3)(g))$.

If g has two disjoint patterns c_1 and c_3, $f_{c_1 \to c_2}$ and $f_{c_3 \to c_4}$ will operate on non-intersected sub-graphs of g. So, $f_{c_1 \to c_2} \circ f_{c_3 \to c_4}(g) = f_{c_3 \to c_4} \circ f_{c_1 \to c_2}(g)$. However, we do not set this restriction to the transformation operation. So, the transformation operation on graphs forms a non-commutative group (non-Abelian group).

Definition 7 Let f be an iteration of transformations $f_1 \circ \dots f_n$, g be a graph. Define sequence $\{g_i\}$ as follows:

$$g_1 = g, \tag{1}$$

$$g_{i+1} = f(g_i). \tag{2}$$

If $g_{n+1} = g_n = f^{n-1}(g_1)$, g_n is the fixed point of graph g under f.

We call $f_{c_1 \to c_2}$ a em zooming-out transformation, if $node(c_1) \supset node(c_2)$. $f_{c_1 \to c_2}$ is *applicable* for graph g, if $g(c_1)$ is not empty.

Definition 8 Let f be an iteration of transformations $f_1 \circ \dots f_n$, where f_i is zooming out. g be a language dependency graph. The spatial dependency graph (SDG) is the fixed point of g under f, written as $\mathcal{S}(g)$.

We need to prove the existence and the uniqueness of $\mathcal{S}(g)$.

Theorem 2 Let f be an iteration of transformations $f_1 \circ \dots \circ f_n$, where f_i is zooming out. For any graph g, there is an m such that $f^m(g)$ is a fixed point.

Proof 2 If f is not applicable for g, then $g = f(g)$. Otherwise let f_m be the first transformation in $f_1 \circ \dots \circ f_n$ which is applicable for g. Let $g' = f_m(g)$, as f_m is zooming out, f_m is not applicable for g'. So, after a maximum of n iterations, there will be no f_i applicable for $f^{n-1}(g)$. So, $f \circ f^{n-1}(g) = f^{n-1}(g)$.

Theorem 3 Let f be an iteration of transformations $f_1 \circ \dots \circ f_n$, where each f_i is zooming out. For any graph g, there is only one fixed point under f.

Proof 3 Suppose there are two different fixed points: $g_{p+1} = g_p = f(g_{p-1})$, $g_{q+1} = g_q = f(g_{q-1})$, and $p > q$. It is obvious that $g_p = f(g_{p-1}) = f^2(g_{p-2}) = \dots = f^{p-q-1}(g_{q+1}) = f^{p-q-2} \circ f(g_{q+1}) = f^{p-q-2} \circ f(g_q) = \dots = g_q$.

The construction of spatial dependency graph can be understood as a process of repeatedly taking-off syntax parts of *cue*s in the way of moving them into edges, resulting in a graph of relations among content words [31]. Proposed the Schematization Similarity Conjecture as follows: *to the extent that space is schematized similarly in language and cognition, language will be successful in conveying space*. Put their conjecture in the cross-linguistic setting – if language is successful in conveying space, descriptions in different languages about the same space shall share some similarity in content and structure, and their semantic representation shall share more similarity after removing syntactic cloths. We call it the *cross-linguistic similarity conjecture*, which is used to examine the soundness of spatial dependency graph. In the next section, we will introduce a method to measure the similarity between dependency graphs.

3 Similarity Between Directed Graphs

Several methods in the literature address the similarity measurement of graphs. One method is purely based on structures: two graphs are similar if they are isomorphic, or one is isomorphic to a sub-graph of the other, or they have isomorphic sub-graphs, [26]. This isomorph-based method is too strong for our case, as most of LDGs of Chinese and German sentences shall belong to the non-isomorphic case – no sub-graphs of their LDGs are isomorphic.

A weaker version is to compute a *graph edit distance* – the similarity between two graphs is measured by minimal cost of transforming one of the two graphs into the other. Basic graph operations shall be defined, such as adding/deleting/substituting nodes/edges, or reversion of edges.

Each operation has an associated cost. Sequences of operations are searched to match one graph to the other. Both of the above methods set priority to the structural information, however, in our task setting, word-to-word matching affects structural matching – if the word x in graph A can only be mapped (translated into) the word y in graph B, the node x must be mapped with node y. In our setting, similarity between graphs is related with contents of nodes – either word-to-word translation suggests mapping of nodes, or mapped nodes affect mapping of unknown words.

A related graph similarity measurement is the SimRank method of [17]. Given two objects, a, b, SimRank method defines the similarity between a, b as follows:

$$s(a,b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s(I_i(a), I_j(b)), \tag{3}$$

where, a, b are nodes of graph G_1 and graph G_2, respectively (G_1 and G_2 can be the same). $C \in (0, 1]$ is a constant, $I(x)$ is the set of in-neighbors of object x. The similarity between a and b is determined by similarities of its in-neighbors. If a or b does not have in-neighbors, $s(a, b)$ is defined as 0. The SimRank algorithm iterates many times over the nodes of G_1 and G_2, to compute the similarity values between the nodes, till these similarity values converge.

The SimRank method meets our setting for reasons as follow: (1) this method considers both content information of nodes and structural information of graphs; (2) it allows empty content information, and uses structural information to approximate similarities. The SimRank method does not meet our task setting in following aspects: (i) it sets the original similarity value to zero, if one of the nodes has no parent node; while in our task setting, we can use a word-to-word translation dictionary to set the original similarity value; (ii) it only considers in-neighbor nodes to make approximation, which is reasonable for the task setting of web-page similarity; while in our setting, similarity between nodes also affected by similarities of out-neighbors; (iii) it computes similarities between nodes of two graphs; while our task is to evaluate the similarity between two graphs. We need to somehow modify the SimRank method, to fit our task setting.

Let G_1 have N nodes $a_1, \ldots, a_N$, G_2 have M nodes $b_1, \ldots, b_M$; $node(G)$ be the set of all nodes of G; $init(a_i, b_j)$ is the in initial similarity value between a_i, b_j; $word(x)$ be a function which returns the word form of node x; $pos(x)$ be

a function which returns the parts of speech of the word of node x; $DICT$ be a word-to-word translation dictionary from language L_1 into language L_2, that is, $DICT = \{(\text{word}^1_{L_1}, \text{word}^1_{L_2}), (\text{word}^2_{L_1}, \text{word}^2_{L_2}), \dots\}$.

We modify the SimRank method into our task setting in following aspects: (a) The original similarity value between two nodes is set to 1, if words of the two nodes exist in a word-to-word translation dictionary, otherwise set to 0:

$$init(a_i, b_j) = \begin{cases} 1, & \text{if } (a_i, b_j) \in DICT, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

(b) The Similarity value between two nodes, $s(a, b)$, are updated by considering similarity values of their both in-neighbors and out-neighbors, and the compatibility between their parts of speeches — we will multiply the sum by a constant value ($\lambda_{pos} \in (0, 1)$). λ_{pos} is understood as the confidence parameter for the similarity between two unknown words, if we only know their parts of speeches. $\lambda_{pos} = 0.36$ is used in this experiment:

$$s_I(a, b) = \begin{cases} \dfrac{C \sum\limits_{i=1}^{|I(a)|} \sum\limits_{j=1}^{|I(b)|} s(I_i(a), I_j(b))}{|I(a)||I(b)|}, & \text{if } |I(a)| * |I(b)| > 0, \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

$$s_O(a, b) = \begin{cases} \dfrac{C \sum\limits_{i=1}^{|O(a)|} \sum\limits_{j=1}^{|O(b)|} s(O_i(a), O_j(b))}{|O(a)||O(b)|}, & \text{if } |O(a)| * |O(b)| > 0, \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

$$s(a, b) = \begin{cases} \lambda_{pos} \frac{s_I(a,b) + s_O(a,b)}{2}, & \text{if } s(a, b) < 1 \wedge pos(a) = pos(b), \\ 0, & \text{if } s(a, b) < 1 \wedge pos(a) \neq pos(b), \\ 1, & \text{otherwise.} \end{cases} \tag{7}$$

(c) The similarity between two graphs is computed by averaging similarity values between nodes:

$$s(G_1, G_2) = \frac{\zeta \sum\limits_{a \in node(G_1)} \sum\limits_{b \in node(G_2)} s(a, b)}{|node(G_1)||node(G_2)|}. \tag{8}$$

ζ is a normalization constant. $\zeta = 1.5$ is used in this experiment.

4 Experiment and Evaluation

In this section, we describe an experiment to evaluate the similarity conjecture on Chinese-German, two languages with totally different grammatical structures.

We take paired Chinese-German sentences in a grammar book as the test set. Our target is to computationally evaluate whether paired sentences have more similar SDGs than their LDGs.

Table 1. RDF prefix of Chinese dependency graphs.

`@prefix`	`ch:`	<http://lin.ch2de.org/ch/>
`@prefix`	`word:`	<http://lin.ch2de.org/ch/word/>

Fig. 4. A fragment of an LDG, where the main verb is taken as the predicate in the RDF representation.

4.1 Representing Graph in the RDF Format

To understand a text, we may need to interlink relations among words in neighborhood sentences, or need to integrate some background knowledge. This encourages us to use the RDF format, which is designed for interlinking knowledge across internet, and has been widely used in semantic web, e.g., [5, 3, 14]. An LDG contains knowledge of word features (represented as node feature), and knowledge of relations among words (represented as relations between nodes).

For example, in Figure 2(a), "`5:`草坪`:n`" contains three pieces of information as follow: (1) the form of the word: 草坪(lawn), (2) this word is located at the 6th position in the sentence (the first position is denoted by '0'), (3) its parts of speech is noun (`n`). For the RDF representation, we introduce a name stem <http://lin.ch2de.org/ch/>. The RDF form of a Chinese word should start with <http://lin.ch2de.org/ch/word>, and multiply 100 to the position number.

For example, the second word will be named as <http://lin.ch2de.org/ch/word/100>. We introduce <http://lin.ch2de.org/ch/form> as the predicate **has-form**, and <http://lin.ch2de.org/ch/pos> as the predicate **has-pos**. As LDG is represented in the RDF format, graph transformation will be carried out by SPARQL queries, e.g., [11]. Spatial dependency graph is the fixed-point of a linguistic dependency graph after a sequence of graph transformations. The RDF prefix is listed in Table 1.

An LDG is transformed into a set of RDF triples, each has the subject-predicate-object format. As we are aiming at semantic representation, we follow the approach used in the frame-net, i.e. [13], [12], and view verbs as semantic schema, which has category roles. For example, the `lie` schema has the actor role, which should be in the subcategory of living things. In Fig 4, the verb 躺(lie) is taken as the predicate with two arguments 他(he) and `SBV` (subject). Formally, we have:

$$\mathbf{word\_100}(\mathbf{word\_0}, \texttt{SBV}), \tag{9}$$

and the RDF form:

$$\texttt{word:0}\quad \texttt{word:100}\quad \texttt{rel:SBV}. \tag{10}$$

Table 2 illustrates a part of the RDF representation of Figure 2(a).

Table 2. RDF representation of the dependency graph in Figure 2 (a).

@prefix	ch	<http://lin.ch2de.org/ch/>
@prefix	word	<http://lin.ch2de.org/ch/word>
@prefix	rel	<http://lin.ch2de.org/ch/rel>
word:0	ch:form	他 #he
word:0	ch:pos	r
word:0	word:100	rel:SBV
word:100	ch:form	躺 #lie
word:100	ch:pos	v
word:100	word:dummy	ROOT

```
prefix de:      <http://lin.ch2de.org/de/>
prefix word:    <http://lin.ch2de.org/de/word/>
prefix rel:     <http://lin.ch2de.org/de/rel:>

ASK {
 ?n ?fuehren ?obj0 .
 ?fuehren de:base "fuehren" .
 ?regie ?n rel:app .
 ?regie de:base "Regie".
}
```

Listing 1.1. SPARQL query to fixed phrase pattern.

4.2 Rule-based Transformation Using SPARQL Queries

The advantage of using the RDF format is that graph transformation can be carried out by SPARQL queries. The basic operation is to search for a sub-graph pattern, and update this sub-graph, if found. Three kinds of sub-graphs can be distinguished: (1) sub-graphs for idioms and fixed expressions; (2) sub-graphs for grammatical rules; (3) sub-graphs containing nodes of functional words (f-node).

As each language has limited number of functional words, we have limited number of sub-graph patterns. Fixed phrases in one language may be expressed by just a word in another language. For example, the fixed phrase *Regie . . . führen* (serve as director) in German is expressed by one word 导演(direct) in Chinese. This pattern of fixed phrase can also be represented by SPARQL query as illustrated in List 1.1. We group components of fixed phrases together, and set the `rel:buddy` relation among them. For the above example, node *Regie* shall be moved to be a child of node *führen* with the relation `rel:buddy`, as illustrated in Figure 5.

The present research focuses on sub-graphs in (2) and (3). We went through all paired Chinese-German sentences in a Chinese grammar book for German speakers, i.e. [29], and manually designed 17 sub-graph patterns for the German sentences, and 11 sub-graph patterns for the Chinese sentences.

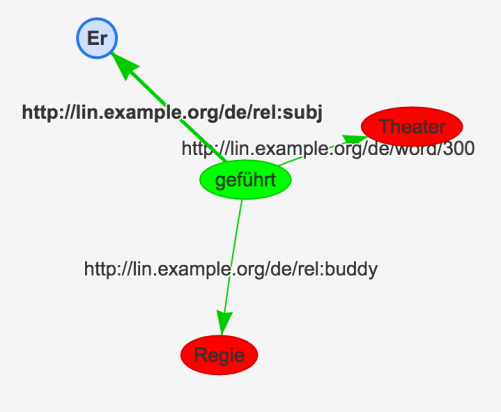


Fig. 5. The fixed phrase *Regie . . . führen* is identified and updated with a `rel:buddy` relation between node *Regie* and node *führen*.

4.3 Experiment Results and Analysis

Experiments are carried out by comparing the similarity between LDGs with the similarity of SDGs of 682 paired Chinese-German sentences in a Chinese grammar book for native German speakers, [29]. For Chinese sentences, we use LTP[1], i.e. [7]; for German sentences, we adopt Parzu[2], i.e. [27].

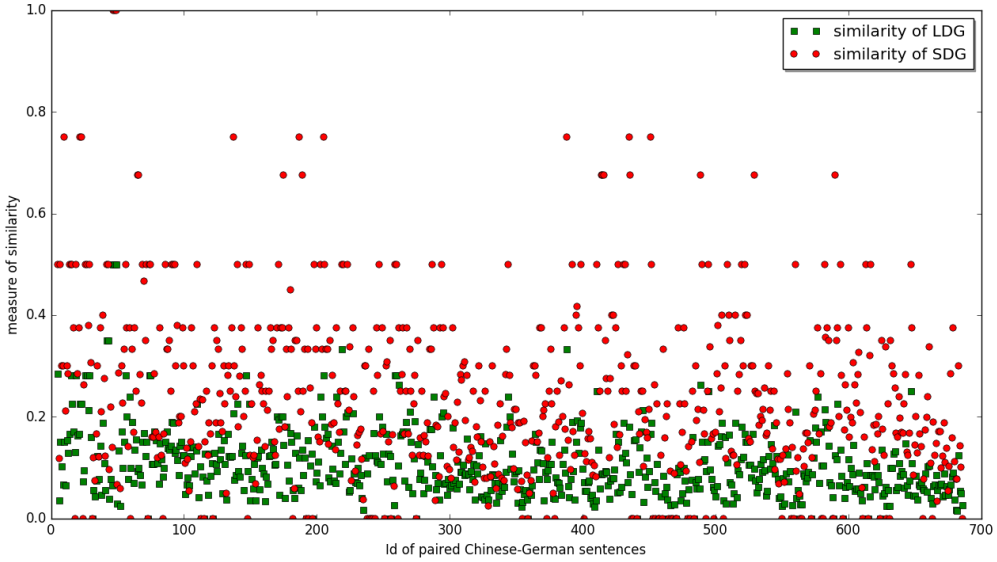
The result is illustrated in Figure 6: the SDGs similarity measures of 610 sentences are higher than the similarity of their LDGs. That is, 89.4% of the paired sentences support our hypothesis that SDGs of paired sentences shall be more similar than their LDGs. We examined the rest 10.6% paired sentences, and found that these 66 paired sentences have SDG similarity value 0.

That is, our dictionary does not have word-to-word translation of the content words for these 66 paired sentences, while functional words in paired LDGs appear in thedictionary. If we remove these 66 invalid inputs, our hypothesis will be supported by 610/(682-66) = 99.03% observations. The ratio between SDG similarity and LDG similarity is illustrated in Figure 6 (b).

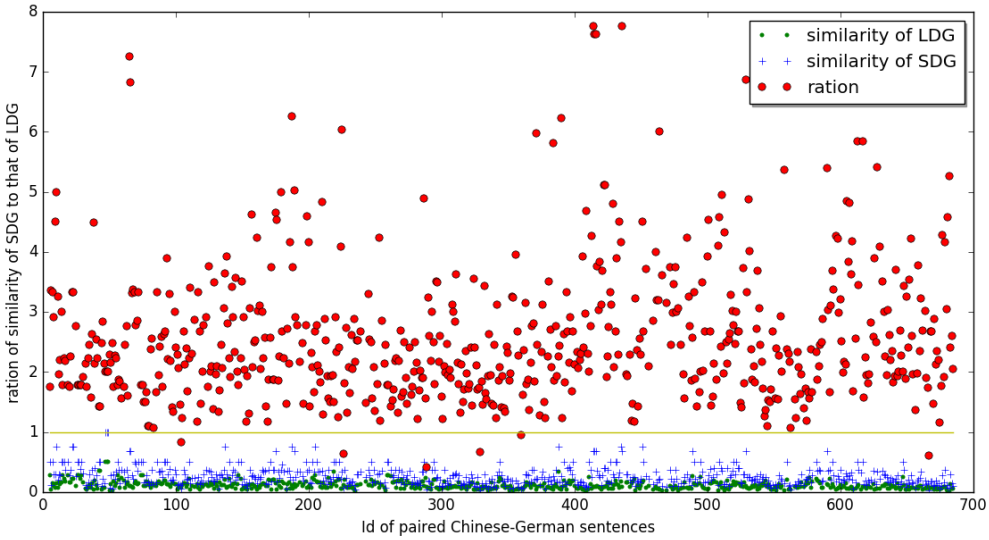
4.4 Error Analysis

There are six paired sentences, as listed in Table 3, whose similarity measures of the LDGs are larger than the non-zero similarity measures of their SDGs. Two main reasons are: (1) word-to-word level mapping are not found at dictionaries: for example, '5月'('May') and '1日('the first') do not have their German translation 'Mai' and '1.' in the current dictionary we used; '放假' ('have vacation') does not have the German translation 'frei haben' ('have free'); (2) SDGs of paired sentences do have different structures, based on the current transformation method – in this experiment, we do not use sub-graph patterns of idioms and fixed phrases.

[1] http://www.ltp-cloud.com/
[2] http://kitt.cl.uzh.ch/kitt/parzu/



(a) Similarity comparison between LDGS and SDGs of paired Chinese-German sentences.



(b) Ratio of SDG similarity to LDG similarity.

Fig. 6. Similarity comparison after removing 66 non-valid observations. Below $y = 1$, there are six red points, which mean that similarity of LDGs is greater than the similarity of SDGs.

Table 3. List of the 6 paired sentences whose LDG similarity is larger than SDG similarity.

Id	Chinese sentence	German sentence
104	有一天晚上突然下了一场大雪。	Eins Abends hat es plötzlich stark geschneit.
226	这件事他几乎不知道。	Sie weiß fast nicht darüber.
289	我们这里从5月1日到5月7日放假。	Wir haben hier vom 1. Mai bis zum 7. Mai frei.
329	由于空气污染严重， 人民难得见到蓝天和白云。	Wegen der schlimmen Luftverschmutzung sieht man selten blauen Himmel und weiße Wolken.
360	尽管这是一个小城， 但是交通很方便。	Obwohl diese Stadt klein ist, sind die Verkehrsverbindungen sehr gut.
667	他身体还没有完全恢复， 不过好多了。	Er ist noch nicht ganz gesund, aber es geht ihm viel besser.

5 Related Work

Our work is related with the research on semantic dependency graph, e.g., [15, 1]. The main difference is that our work uses node information of LDG as edge lables in SDG, instead of introducing semantic labels at the very beginning. Our method introduces the flexibility in semantic analysis: under concrete contexts or demands, we can introduce external semantic information, for example, from [15], or some emotional categories.

The idea of updating a structure until a fixed point is not new in the NLP literature. Many rule-based machine translation systems used this idea in grammar rewriting, e.g., [24, 6, 16]. Our work can be understood as rewriting rules into the direction of semantic representation. Most work on semantic representation is relied on manual annotation. An ambitious and popular semantic representation is the Abstract Meaning Representation (AMR), i.e. [4], and Chinese AMR is also recently released[3].

[15] is a semantic representation of German, which is based on a large set of over 10-years' long hand-annotated semantic dictionary. [25]'s Proposition Bank trained semantic-roles based on annotated corpus of Penn Treebank. [8] presents a unified neural NLP system for four common tasks: POS tagging, Chunk, name entity recognition, and semantic role labelling (SRL). The SRL task was trained by using convolution neural network.

Semantic representation is normally carried out as a supervised machine learning task: To get the meaning representation of a form, we first need to have meaning representations of other forms (by manual annotation), establish some (statistical or neural) relations between them, and apply these relations for the new form to get its meaning representation.

Different from the above work, this work proposed in this paper attempts to approach to semantic representation using linguistic cues, instead of annotated semantic corpus. The small scale experiment shows SDGs of paired Chinese-German sentences are more similar than their LDGs.

[3] http://www.cs.brandeis.edu/ clp/camr/camr.html

Our on-going work is to experiment with large data-sets, i.e. English-Chinese, English-German paired corpus used in machine translation.

6 Application of Spatial Dependency Graphs in Word-embedding, and Machine Translation

Word-embedding plays a key-role in neural-based NLP. [19] shows that word-embedding based on dependency relation outperforms the word-embedding based on neighborhood-relations in the raw corpus, i.e. [23], for the reason that semantic-relations between words are more precisely captured in the dependency graph than in the literal sequence. Following this reason, the word-embedding based SDG shall out-perform the word-embedding based on raw dependency graph.

The second application of SDG is in Cue-based Machine Translation, as introduced in [10]. As SDG is acquired based on cues of a language, the SDG representation of a pair sentence approximates an aggregated meaning representation. The alignment between components of paired SDGs shall outperform that of paired raw dependency graph, i.e. [28].

Acknowledgments. We are indebted to anonymous reviewers for their critical and valuable comments. Partial financial support from NSFC under Grant No. 61472177 is greatly acknowledged.

References

1. Agić, Ž., Koller, A., Oepen, S.: Semantic dependency graph parsing using tree approximations. In: Proceedings of the 11th International Conference on Computational Semantics, pp. 217–227 (2015)
2. Alexandrescu, A., Kirchhoff, K.: Graph-based learning for statistical machine translation. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 119–127 (2009)
3. Allemang, D., Hendler, J.: Semantic web for the working ontologist: Effective modeling in RDFS and OWL. Morgan Kaufmann Publishers Inc., Elsevier (2011) doi: 10.1016/C2010-0-68657-3
4. Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., Schneider, N.: Abstract meaning representation for sembanking. In: Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse, pp. 178–186 (2013)
5. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American, vol. 284, no. 5, pp. 34–43 (2001)
6. Boitet, C.: The french national MT-project: Technical organization and translation results of CALLIOPE-AERO. Computers and Translation, vol. 1, no. 4, pp. 239–267 (1986) doi: 10.1007/BF00936424
7. Che, W., Li, Z., Liu, T.: LTP: A Chinese language technology platform. In: Coling 2010: Demonstrations, pp. 13–16 (2010)
8. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research, vol. 12, pp. 2493–2537 (2011)

9. De-Marneffe, M. C., Grenager, T., MacCartney, B., Cer, D., Ramage, D., Kiddon, C., Manning, C. D.: Aligning semantic graphs for textual inference and machine reading. In: Proceedings of the AAAI Spring Symposium at Stanford, pp. 658-476 (2007)
10. Dong, T., Cremers, A. B.: A novel machine translation method for learning Chinese as a foreign language. In: International Conference on Intelligent Text Processing and Computational Linguistics, vol. 8404, pp. 343–354 (2014) doi: 10.1007/978-3-642-54903-8_29
11. DuCharme, B.: Learning SPARQL: Querying and updating with SPARQL 1.1. O'Reilly Media, Inc (2013)
12. Feldman, J.: From molecule to metaphor: A neural theory of language. The MIT Press (2006) doi: 10.7551/mitpress/3135.001.0001
13. Fillmore, C. J.: Scenes-and-frames semantics: Linguistic structures processing. Fundamental Studies in Computer Science, North Holland Publishing, vol. 5, no. 59, pp. 55–88 (1977)
14. Grobe, M.: RDF, Jena, SparQL and the semantic web. In: Proceedings of the 37th annual ACM SIGUCCS Fall Conference: Communication and Collaboration, pp. 131–138 (2009) doi: 10.1145/1629501.1629525
15. Helbig, H.: Knowledge representation and the semantics of natural language. Springer (2006) doi: 10.1007/3-540-29966-1
16. Nakamura, J. I., Tsujii, J. I., Nagao, M.: GRADE: A software environment for machine translation. Computers and Translation, vol. 3, no. 1, pp. 69–82 (1988) doi: 10.1007/BF02057968
17. Jeh, G., Widom, J.: SimRank: A measure of structural-context similarity. In: Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Minings, pp. 538–543 (2002) doi: 10.1145/775047.775126
18. Klein, D.: The unsupervised learning of natural language structure. PhD thesis, Computer Science Department, Stanford University (2005)
19. Levy, O., Goldberg, Y.: Dependency-based word embeddings. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, vol. 2, pp. 302–308 (2014) doi:10.3115/v1/P14-2050
20. Li, L., Way, A., Liu, Q.: Graph-based translation via graph segmentation. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, vol. 1, pp. 97–107 (2016) doi: 10.18653/v1/P16-1010
21. Li, P., Bates, E., MacWhinney, B.: Processing a language without inflections: A reaction time study of sentence interpretation in Chinese. Journal of Memory and Language, vol. 32, no. 2, pp. 169–192 (1993) doi: 0.1006/jmla.1993.1010
22. MacWhinney, B., Bates, E., Kliegl, R.: Cue validity and sentence interpretation in English, German, and Italian. Journal of Verbal Learning and Verbal Behavior, vol. 23, no. 2, pp. 127–150 (1984) doi: 10.1016/S0022-5371(84)90093-8
23. Mikolov, T., Yih, W. T., Zweig, G.: Linguistic regularities in continuous space word representations. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 746–751 (2013)
24. Nakamura, J. I., Tsujii, J. I., Nagao, M.: Grammar writing system (GRADE) of Mu-Machine translation project and its characteristics. In: 10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics, pp. 338–343 (1984) doi: 10.3115/980491.980560
25. Palmer, M., Gildea, D., Kingsbury, P.: The proposition bank: An annotated corpus of semantic roles. Computational linguistics, vol. 31, no. 1, pp. 71–106 (2005) doi: 10.1162/0891201053630264
26. Pelillo, M.: Replicator equations, maximal cliques, and graph isomorphism. Neural Computation, vol. 11, no. 8, pp. 1933–1955 (1999) doi: 10.1162/089976699300016034

27. Sennrich, R., Schneider, G., Volk, M., Warin, M.: A new hybrid dependency parser for German. In: Proceedings of the Biennial GSCL Conference, pp. 115–124 (2009) doi: 10.5167/uzh-25506
28. Shen, L., Xu, J., Weischedel, R.: String-to-dependency statistical machine translation. Computational Linguistics, vol. 36, no. 4, pp. 649–671 (2010) doi: 10.1162/coli_a_00015
29. Song, J.: PONS grammatik chinesisch. PONS GmbH (2007)
30. Talmy, L.: How language structures space. Spatial Orientation, pp. 225–281 (1983) doi: 10.1007/978-1-4615-9325-6_11
31. Tversky, B., Lee, P. U.: How space structures language. Spatial Cognition, vol. 1404, pp. 157–175 (1998) doi: 10.1007/3-540-69342-4_8

ISSN 1870-4069

Sentiment Identification in Code-mixed Social Media Text

Souvick Ghosh[1], Satanu Ghosh[2], Dipankar Das[3]

[1] Rutgers University,
United States

[2] MAKAUT,
India

[3] Jadavpur University,
India

{satanu.ghosh.94, dipankar.dipnil2005}@gmail.com
souvick.ghosh@rutgers.edu

Abstract. Sentiment analysis is the Natural Language Processing (NLP) task dealing with the detection and classification of sentiments in texts. While some tasks deal with identifying presence of sentiment in text (Subjectivity analysis), other tasks aim at determining the polarity of the text categorizing them as *positive*, *negative* and *neutral*. Whenever there is presence of sentiment in text, it has a source (people, group of people or any entity) and the sentiment is directed towards some entity, object, event or person. Sentiment analysis tasks aim to determine the subject, the target and the polarity or valence of the sentiment. In our work, we try to automatically extract sentiment (positive or negative) from Facebook posts using a machine learning approach. While some works have been done in code-mixed social media data and in sentiment analysis separately, our work is the first attempt (as of now) which aims at performing sentiment analysis of code-mixed social media text. We have used extensive pre-processing to remove noise from raw text. Multilayer Perceptron model has been used to determine the polarity of the sentiment. We have also developed the corpus for this task by manually labelling Facebook posts with their associated sentiments.

Keywords: Sentiment in the text, perceptron, generalized Delta rule, social media text.

1 Introduction

Sentiment analysis - of social media in particular - has become a popular area of research in present times. The massive proliferation of social media has been a catalyst in this regard. A culture shift can be noticed where the users comfortably and candidly express their emotions, opinions or sentiments online. This has encouraged the researchers to analyze and study the presence of sentiments from social media.

Extraction of sentiment from social media - like Facebook or microposts like Twitter-can serve a myriad of purposes. These texts often express opinion about a variety of topics.

It can be the appraisal of the user about certain products or incidents, the state of mind of the speaker or any intended emotional communication that he may want to have with potential readers.

User reviews on e-commerce sites, opinions on web blogs, tweets[1] and Facebook[2] posts, can be mined for assessing polarity of opinion. Businesses use the power of text analytics behind their data mining technology. Sentiment analysis helps businesses in advertising, marketing and making business decisions for better customer satisfaction. Organizations can determine public opinion about their products and services.

Similarly, consumers can use sentiment analysis while researching products prior to purchase. It can also be used to investigate the web for forecasting electoral results (by evaluating voter sentiment) and track political preferences. Recently, social media analysis has been used extensively to identify cyber-bullying prevalent in the web space [26].

Although we have come across various tasks conducted on multilingual texts, the task of sentiment analysis, in particular, has not been explored for multilingual code-mixed texts. Researches have focused on identifying the word-level language and parts-of-speech of code-mixed text [34, 35, 15, 14, 12, 39].

They conclude that this type of text differs significantly from traditional English texts and needs to be processed differently. However, different forms of texts require different methods for sentiment analysis. For example, if we look at sentiments in scientific papers, it is hedged and indirect while the sentiments are more direct in movie or product reviews. Traditional texts like reviews and newspaper are structured and follow a definite pattern. Also, the writing is more formal and composed. Social media texts on the other hand are largely informal. They are concise and informal with several linguistic differences.

In our work, we have used code-mixed social media data which have been collected from Facebook post. The text is informal and conversational in accordance with social media characteristics. It is mostly bilingual though the presence of three languages in a single post is not entirely uncommon in our data. Initially, we pre-process the text to normalize the irregular words. We also remove noise from the text prior to processing it and translate the abbreviations to regular words wherever applicable.

We label the posts with their respective part-of-speech tags. Traditionally, sentiment classifiers show improvements by using part-of-speech features. We make use of various word-level, dictionary-based and stylistics features relevant to social media text to classify the sentiment as subjective or objective. Subjective posts are further categorized as positive or negative in polarity. We use various machine learning algorithms for our final classification. Artificial neural network model performs best in our experiments.

The remainder of this paper is structured as follows: Section 2 gives an overview of the background and related work. In Section 3, we present the dataset. The working model for our system is described in Section 4. We describe in detail the pre-processing and feature selection used to build the classification models. In Section 5, we present the results obtained using different combinations of features.

[1] twitter.com

[2] www.facebook.com

We evaluate the performance of various machine learning models that we used in our experimentation. Section 6 summarizes the main findings of this work and sketches the lines for future work.

2 Related Work

Research regarding emotion and mood analysis in text - is becoming more common recently, in part due to the availability of new sources of subjective information on the web. The work of Ortony et al. [28] was one of the very first in the area of sentiment classification. They focused on the actual taxonomy and isolation of terms with an emotional connotation.

Identifying the semantic polarity (positive vs. negative connotation) of words has been done using different approaches. Some of the works (knowledge-based) explicitly attempted to find features indicating that subjective language is being used. Hatzivassiloglou and McKeown [19] made use of corpus statistics, Wiebe [41] used linguistic tools such as WordNet [23], and Liu et al. [24] used lexicon-based classifier. Turney and Littman [37] work on classification of reviews was based on using an unsupervised learning technique.

They found the mutual information between document phrases and the words like "excellent" and "poor". The mutual information was computed using statistics gathered by a search engine. In their work on automatic classification of sentiment in online domains, Pang et al. [30] evaluated the performance of different classifiers on movie reviews. They demonstrated that that standard machine learning techniques outperform human-produced baselines.

Typically, methods for sentiment analysis produce lists of words with polarity values assigned to each of them. This method has been successfully employed for applications such as product review analysis and opinion mining [6, 7, 17, 30, 27, 38, 11]. Holzman and Pottenger [20] reported high accuracy in classifying emotions in online chat conversations by using the phonemes extracted from a voice-reconstruction of the conversations.

Rubin et al. [33] investigated discriminating terms for emotion detection in short text while Read [31] described a system for identifying affect in short fiction stories, using the statistical association level between words in the text and a set of keywords. In another work, Read [32] used distant supervision to build the corpus.

There has been some work by researchers in the area of phrase level and sentence level sentiment classification [42] and on analyzing blog posts [25]. Wilson et al. [42] determined whether an expression is neutral or polar and then disambiguated the polarity of the polar expressions. With this approach, their system was able to automatically identify the contextual polarity for a large subset of sentiment expressions.

Sentiment analysis of social media text has received a lot of interest from the research community in the recent years with the rise to prominence of Facebook and Twitter. Ding et al. [10] used context-dependent sentiment words in their work and Tan et al. [36] suggested combining learning-based and lexicon-based techniques using a centroid classifier.

Table 1. Statistics of the corpus.

Language Tags	Number Of Words Present	Percentage Of Corpus
English (En)	9988	52.72
Bengali (Bn)	8330	43.97
Hindi (Hi)	626	3.3

Go et al. [16] used positive and negative emoticons to classify tweet polarity. They showed that machine learning algorithms (Naive Bayes, Maximum Entropy, and SVM) have accuracy above 80% when trained with emoticon data. Paroubek [29] showed how to automatically collect a corpus for sentiment analysis and opinion mining purposes. They concluded that authors use syntactic structures to describe emotions or state facts and some POS-tags may be strong indicators of emotional text. They obtained best results using Naive Bayes classifier that uses N-gram and POS-tags as features.

Diakopoulos and shamma [9] used crowd-sourcing techniques to manually rate polarity in Twitter posts. In their work, De Choudhury et al. [8] classified human affective states from posts shared on Twitter. Wang and Manning [40] highlighted the suitability of Support Vector Machine or Naive Bayes for different domains. Our approach is similar to that of Zhang et al. [43] and Ghosh et al. [13] who presented the idea of ternary classification system (positive, negative and neutral).

They used target words bearing sentiment and supervised learning for classification. We also use some techniques for noise reduction which was inspired by Hu et al. [21]. They proposed building a sophisticated feature space to handle noisy and short messages in their work on Twitter sentiment analysis.

3 Dataset

A recent shared task was conducted by Twelfth International Conference on Natural Language Processing (ICON-2015)[3] , for part-of-speech tagging of transliterated social media text. For the shared task in that corpus, data was collected from Bengali-English Facebook chat groups. The Facebook posts are in mixed English-Bengali and English-Hindi - and have been obtained from the ”JU Confession” Facebook group, which contains posts in English-Bengali with few Hindi words in some cases.

We have modified the ICON Shared Task Corpora for our work on sentiment analysis. The dataset contains three languages - Bengali, Hindi and English. The data set contains 882 posts in total. The statistics for the dataset have been presented in Table 1.

The purpose of the implementation is to be able to automatically classify a post as a positive or negative tweet sentiment wise. The classifier needs to be trained and to do that we needed a list of manually classified posts. We used 2 annotators to classify the posts into three categories - positive, negative or neutral.

We have calculated Kappa co-efficient to measure the inter-annotator agreement. Kappa co-efficient is a reliable and robust measure to measure the agreement between two users. It takes into account the agreement occurring by chance and hence, is more useful than percent agreement calculation.

[3] ltrc.iiit.ac.in/icon2015/contests.php

Table 2. Inter-annotator agreement.

Annotator 1	**Annotator 2**			
	Positive	Neutral	Negative	Total
Positive	200	146	13	359
Neutral	46	268	26	340
Negative	6	80	97	183
Total	252	494	136	

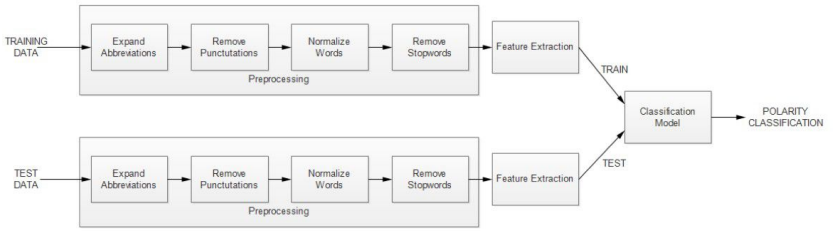


Fig. 1. Overview of the system architecture.

For the above data, po is 0.641 and pe is 0.3642, therefore giving a Kappa co-efficient of 0.4354. Because the Kappa measure is low, so we have obtained the instances where the annotators are unanimous about the sentiment polarity. There are a total of 565 such instances. We have used these posts for our sentiment polarity classification.

4 System Description

The process of sentiment analysis can be divided into three major parts : pre-processing of raw posts, feature identification and extraction and finally, the classification of sentiment as positive, neutral or negative. The steps have been discussed in sequential order.

4.1 Pre-processing of the Facebook Posts

The following steps were performed to pre-process the raw posts prior to feature extraction.

1. **Expansion of Abbreviations**

   As social media text is often non-traditional and informal in nature, the posts had to be pre-processed initially to remove noise. We have used an abbreviation list to normalize all the words that were abbreviated. For example, *btw* was replaced by *"by the way"*, *clg* by *"college"*, *hw* by *"how"* and so on.

2. **Removal of Punctuations**

   Before processing the post any further, we remove all punctuations from the text. Mostly social media texts contains a lot of punctuations and their usage is

often arbitrary in nature, not adhering to grammatical norms. To compound the problem further, punctuations like stop, question mark and exclamation marks are often used multiple times in succession. By removing all the punctuations, we try to make our text as noiseless as possible. We keep a record of the number of different punctuations in the text which has been used as a feature for classification.

3. **Removal of Multiple Character Repetitions**

   It is often found in social media text that certain characters are repeated more than once. These non-conformational spellings are very hard to deal with as they cannot be successfully matched to any dictionary. For example, *lol* (abbreviated form of *laughing out loud*)can be written as *loool*, *looool* or *looooooool*. We use pre-processing in order to reduce all these occurrences to lool. Any character which occurs more than two times in a row is replaced by two occurrences of the same character. Some other examples are *ahhhh* (reduced to *ahh*) and *uhhhh* (reduced to *uhh*). However, we maintain a record of the number of repetitions as this could be used by the author in specific situations to reflect sentiment.

4.2 Feature Extraction

In our work, we used the following features to train our machine learning model.

1. **Number of Word Matches with Sentiwordnet (SWN):** We have used SentiWordNet[4] as one of the sentiment resources. SWN is a lexical resource for sentiment analysis. It assigns three sentiment scores positivity, negativity and objectivity to each synset of WordNet. So, a given word can have a positive or negative score or both. We have extracted all the positive and negative words from SWN. The final list contains 17027 positive words and 17992 negative words. For a given data instance or sentence, we find if the normalized words are a match with any words in these two lists. In a sentence we count the number of words which matches with the positive word list and the number of words which matches with the negative word list and the assign the difference between the positive and negative word count as a feature.
2. **Number of Word Matches with Opinion Lexicon (OL):** Similar to SentiWordNet, Opinion Lexicon[5] is another lexical resource for sentiment analysis. It contains a list of positive and negative opinion words or sentiment words for English. There is a total of 2006 positive words and 4783 negative words. We find the number of matches to both the lists and the difference is taken as our second feature.
3. **Number of Word Matches with English Sentiment Words (ESW):** We have collected a list of positive and negative words from the internet for sentiment classification. We hand-labeled a few words in the training data as root words which depicted emotion. Using bootstrapping, we expanded this list of words. It contains 3075 positive words and 4003 negative words. This list concentrates more on the words which appear in social media context.

[4]http://sentiwordnet.isti.cnr.it/

[5]https://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html

Similar to the previous two features, we find the number of matches to the positive and negative lists and the difference between the two is considered as our third feature.

4. **Number of Word Matches with Bengali Sentiment Words (BSW):** This list was developed to tackle the presence of sentiment in Bengali words. As we are dealing with multilingual text, it was essential to develop this list for Bengali. Das and colleagues [3, 4, 5] developed SentiWordNet for Indian Languages. However, this list contained words in Bengali (or Brahmic) scripts. As we are dealing with transliterated text, this wordlist required transliteration to English. Finally, we developed a positive and a negative wordlist for transliterated Bengali words. The number of words in the positive wordlist is 1778 while the negative wordlist contains 3713 words. The difference in number of matches to both the lists is considered as our next feature.

5. **Number of Colloquial Bengali Sentiment Words (CBW):** We have created this list for Bengali words which often appear in social media text. It must be noted that Bengali Sentiment Words developed previously is more formal in nature and therefore, not sufficient for identifying colloquial words which appear in Facebook posts or Twitter texts. For example, words like *jata* (hopeless), *hebby* (excellent), *phot* (get lost) are not captured by Bengali Sentiment Words. We create two lists – positive and negative wordlists - tries to incorporate all such words which may indicate the presence of sentiment in the text. The number of matches to both the lists is determined and the difference is assigned as feature.

6. **Density of Curse or Bad Words (CW):** We have used a list of curse words (words which are used as bad words in majority instances) developed by Huang et al. [22] in their work on cyberbullying. In their work, the authors collected 713 curse words (e.g., 'asshole', 'bitch' etc.) and hieroglyphs (such as '5hit', '@ss' etc.) based on online resources. We have used this list to find out all the words which have been used with a negative sentiment.

7. **Part-of-Speech Tags (POS):** All the posts were tagged manually for parts-of-speech information. It has been noted that words belonging to certain part-of-speech tags (like JJ, RB and JJ-RB) are usually used to express sentiment. These part-of-speech tags can be considered as features to detect presence of sentiment in commonly occurring unigram and bigrams in the training data.

8. **Number of All Uppercase Words (UW):** Based on the findings of Dadvar et al. [2], capital letters can represent shouting or strong opinion in online chats and posts. We have identified the number of words in a post which are written in all capital letters. This is used as a feature to detect the presence of emotion or sentiment in online settings.

9. **Density of Exclamation Points (E):** Just like the uppercase letters, exclamation points also stand as emotional comments. To identify strong emotions in social media context, we chose the number of exclamation points as a feature for our model.

The number of exclamation points is normalized by the number of words present in the text.

10. **Density of Question Marks (Q):** Similar to the last feature, multiple question marks in the text can denote surprise, excitement or agitation of the user. We chose the number of question marks as our next feature. The number of question marks is normalized by the number of words present in the text.
11. **Number of Character Repetitions in a Word (R):** It is often observed that users tend to repeat a number of characters – vowels or consonants – to stress their opinion in social media conversations. Words like *loool*, *lolzzzz*, *uffffff*, *ahaaa*, *greaaat* are quite common in social media texts. While we reduce all such words during our pre-processing step, we have also maintained a record of all such occurrences. These repetitions are often indicative of sentiment and we use it as one of our feature.
12. **Frequency of Code Switches (CS):** As we are dealing with multilingual texts, we have considered the frequency of code switching as one of our features. It is often observed that the writer shifts language to clarify his opinion. We have tried to exploit this social and communication needs for this language shifting to determine the presence of sentiment. This frequency (number of language switching points) is normalized by the number of words in a particular post.
13. **Number of Smiley Matches (S1 And S2):** Smileys are quite prevalent in social media text and often form a primary way of expressing emotion. We have created two resources for identifying smiley in text. The first one contains 269 positive smileys and 170 negative smileys. The second list contains 243 smileys. We found the number of matches to both the lists and used it as a feature.

4.3 Classification of Sentiment Polarity

We obtain results for the 565 posts for which both the annotators agreed on the polarity. We use 70% of the dataset for training and 30% for testing purposes.We split the dataset using 400 posts for training and 165 posts for testing.

We use the machine learning software WEKA[6] [18]. We combine the above features to form a feature set and employ a number of machine learning algorithms for classification. The best results were produced by Multilayer Perceptron model. This classifier uses back propagation to classify instances into three categories - *positive*, *negative* and *neutral*. The nodes in this network are all sigmoid. The learning rate and momentum rate for the back propagation algorithm was kept at 0.3 and 0.2 respectively.

The number of epochs was set to 500 and the random number generator was seeded using value 0. Individually, none of the features was able to detect positive or negative instances in citation. This is due to the biasness of the system. We perform feature analysis by removing one feature at a time to determine if any feature is more important than the other.

[6]http://www.cs.waikato.ac.nz/ml/weka/downloading.html

Table 3. Impact of adding each feature iteratively to the last.

Feature added	Correct classifications	Incorrect classifications	Accuracy
G1	110	55	0.667
G1 + G2	**113**	**52**	**0.685**
G1 + G2 + G3	101	64	0.612

Table 4. Impact of each feature calculated by eliminating one at a time.

Feature Eliminated	Correct classifications	Incorrect classifications	Accuracy
None	104	61	0.630
SWN	109	56	0.661
OL	103	62	0.624
ESW	102	63	0.618
BSW	104	61	0.630
CBW	101	64	0.612
S	105	60	0.636
POS	**100**	**65**	**0.606**
UW	110	55	0.667
E	107	58	0.649
Q	105	60	0.636
R	107	58	0.649
CS	106	59	0.642
S1	100	65	0.606
S2	106	59	0.642

We also check by adding one feature group at a time. The classification confidence score from WEKA and the number of matches to our citation specific lexicon is used to develop a post-processing algorithm.

5 Results and Observations

For feature analysis, we have grouped the different kind of features and obtained the impact of each group in classification. We have grouped the word (or dictionary) based features into Group 1 (G1), syntactic features into Group 2 (G2) and the style based features into Group 3 (G3).
G1: SWN + OL + ESW + BSW + CBW + CW + S,
G2: POS,
G3: UW + E + Q + R + CS.

From Table 3 it is evident that word based features (Group 1) and syntactic features (Group 2) produce the best results collectively. The accuracy decreases when we include the style based features for classification.

Table 5. Confusion matrix for classification.

	Positive	**Neutral**	**Negative**
Positive	25	23	4
Neutral	10	78	3
Negative	4	8	10

Table 6. Precision, recall and f-measure.

	Precision	**Recall**	**F-measure**
Class Positive	0.641	0.481	0.55
Class Neutral	0.716	0.857	0.78
Class Negative	0.588	0.455	0.513

Table 4 serves to highlight the impact of individual features in classification. At each turn, we eliminate one of the features while keeping all the other features. The accuracy suffers the maximum on elimination of POS (JJ, RB and RB_JJ) features and the polar smiley list. Elimination of all the style based features (UW, E, Q, R and CS) shows improvement in accuracy. This is in accordance to our findings in Table 3. Elimination of SWN also improves accuracy. Removing BSW – which comprises of conformational (or traditional) Bengali words – do not affect accuracy proving the fact that social media text requires tailor made resources.

Table 5 shows the confusion matrix for the polarity classification (using word based and semantic features). The precision, recall and f-measure of the supervised and baseline systems are compared in Table 6.

If we consider the baseline model to contain all the instances of neutral polarity, then we can achieve an accuracy of 55.2%. Our best performing system shows an accuracy of 68.5%. So we can see that our supervised system shows improvement over the baseline model. However, the learning algorithm was slightly biased towards neutral classification which is evident from the confusion matrix.

Most of the errors are due to positive and negative citations being identified as neutral. In future works, we will need to fine tune our classification features so that the system can identify positive and negative citations more efficiently. Also using a larger dataset to train the system would eliminate the bias towards neutral classification of polarity.

6 Conclusion and Future Work

As per our knowledge, there exists no sentiment classifier for code-mixed social media text. We have performed a machine learning based sentiment classification of Facebook posts. The polarity of each post has been classified as *positive*, *negative* and *neutral*. As there has not been any similar work before, we had to create a dataset of our own. Two human annotators classified the polarity of each post. Due to the inherent complexity of social media text, use of arbitrary emoticons and presence of sarcasm, the agreement between the human annotators was quite low with a Kappa co-efficient of 0.4354.

Although the entire dataset consists of 882 posts, we have used only 565 posts where the annotators were unanimous about the polarity of underlying sentiment. We used word-based, semantic and style-based features for classification. The best result was obtained using a combination of word-based and semantic features with an accuracy of 68.5%.

As our dataset is relatively small, we would like to create a larger dataset in future. Sentiment annotation can also be done using distant supervision based on the presence of emoticons. However, such an approach can lead to noisy dataset. Creating a gold standard for all future tasks is a priority for us. In this work, we have not focused on detection of sarcasm in text. Also, we have not handled negation in data. We would like to concentrate on dealing with these issues in our next work.

Apart from that, sentiment classification can be further improved by better handling comparisons and by detecting sentiment targeted towards an entity in particular. Handling of context switches is also important. Developing a real time accurate sentiment classifier model is the ultimate goal which we strive to achieve in future.

References

1. Balahur, A.: Sentiment analysis in social media texts. In: Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 120–128 (2013)
2. Dadvar, M., Trieschnigg, D., Ordelman, R., Jong, F.: Improving cyberbullying detection with user context. European Conference on Information Retrieval, vol. 7814, pp. 693–696 (2013) doi: 10.1007/978-3-642-36973-5_62
3. Das, A., Bandyopadhyay, S.: SentiWordNet for indian languages. In: Proceedings of the Eighth Workshop on Asian Language Resouces, pp. 56–63 (2010)
4. Das, A., Bandyopadhyay, S.: Dr sentiment knows everything! In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Systems Demonstrations, pp. 50–55 (2011)
5. Das, A., Gambäck, B.: Sentimantics: Conceptual spaces for lexical sentiment polarity representation with contextuality. In: Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis, pp. 38–46 (2012)
6. Das, S. R., Chen, M. Y.: Yahoo! for amazon: Sentiment parsing from small talk on the web. Management Science, vol. 53, no. 9, pp. 1375–1388 (2007) doi: 10.1287/mnsc.1070.0704
7. Dave, K., Lawrence, S., Pennock, D. M.: Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In: Proceedings of the 12th international conference on World Wide Web, pp. 519–528 (2003) doi: 10.1145/775152.775226
8. De-Choudhury, M., Gamon, M., Counts, S.: Happy, nervous or surprised? Classification of human affective states in social media. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 6, pp. 435–438 (2012) doi: 10.1609/icwsm.v6i1.14335
9. Diakopoulos, N. A., Shamma, D. A.: Characterizing debate performance via aggregated twitter sentiment. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1195–1198 (2010) doi: 10.1145/1753326.1753504
10. Ding, X., Liu, B., Yu, P. S.: A holistic lexicon-based approach to opinion mining. In: Proceedings of the 2008 International Conference on Web Search and Data Mining, pp. 231–240 (2008) doi: 10.1145/1341531.1341561
11. 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 (LREC'06) (2006)

12. Gella, S., Sharma, J., Bali, K.: Query word labeling and back transliteration for indian languages: Shared task system description. Working Notes - Forum for Information Retrieval Evaluation, vol. 3, pp. 89–105 (2013)
13. Ghosh, S., Das, D., Chakraborty, T.: Determining sentiment in citation text and analyzing its impact on the proposed ranking index. Lecture Notes in Computer Science, vol. 9624, pp. 292–306 (2017) doi: 10.1007/978-3-319-75487-1_23
14. Ghosh, S., Ghosh, S., Das, D.: Labeling of query words using conditional random field. FIRE Workshops, pp. 29–32 (2016)
15. Ghosh, S., Ghosh, S., Das, D.: Part-of-speech tagging of code-mixed social media text. In: Proceedings of the Second Workshop on Computational Approaches to Code Switching, pp. 90–97 (2016) doi: 10.18653/v1/W16-5811
16. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford (2009)
17. Grefenstette, G., Qu, Y., Shanahan, J. G., Evans, D. A.: Coupling niche browsers and affect analysis for an opinion mining application. In: Proceedings of Recherche d'Information Assistée par Ordinateur (RIAO), pp. 186–194 (2004)
18. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H.: The WEKA data mining software: an update. ACM SIGKDD Explorations Newsletter, vol. 11, no. 1, pp. 10–18 (2009) doi: 10.1145/1656274.1656278
19. Hatzivassiloglou, V., McKeown, K.: Predicting the semantic orientation of adjectives. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, pp. 174–181 (1997) doi: 10.3115/976909.979640
20. Holzman, L. E., Pottenger, W. M.: Classification of emotions in internet chat: An application of machine learning using speech phonemes. Retrieved November, vol. 27, no. 2011, pp. 50 (2003)
21. Hu, X., Tang, L., Tang, J., Liu, H.: Exploiting social relations for sentiment analysis in microblogging. In: Proceedings of the sixth ACM international conference on Web search and data mining, pp. 537–546 (2013) doi: 10.1145/2433396.2433465
22. Huang, Q., Singh, V. K., Atrey, P. K.: Cyber bullying detection using social and textual analysis. In: Proceedings of the 3rd International Workshop on Socially-Aware Multimedia, pp. 3–6 (2014) doi: 10.1145/2661126.2661133
23. Kamps, J., Marx, M., Mokken, R. J., Rijke, M.: Using WordNet to measure semantic orientations of adjectives. In: Proceedings of the Fourth International Conference on Language Resources and Evaluation, vol. 4, pp. 1115–1118 (2004)
24. Liu, H., Lieberman, H., Selker, T.: A model of textual affect sensing using real-world knowledge. In: Proceedings of the 8th international conference on Intelligent user interfaces, pp. 125–132 (2003) doi: 10.1145/604045.604067
25. Mishne, G.: Experiments with mood classification in blog posts. In: Proceedings of ACM SIGIR 2005 workshop on stylistic analysis of text for information access, vol. 19, pp. 321–327 (2005)
26. Nahar, V., Unankard, S., Li, X., Pang, C.: Sentiment analysis for effective detection of cyber bullying. Lecture Notes in Computer Science, vol. 7235, pp. 767–774 (2012) doi: 10.1007/978-3-642-29253-8_75
27. Nasukawa, T., Yi, J.: Sentiment analysis: Capturing favorability using natural language processing. In: Proceedings of the 2nd international conference on Knowledge capture, pp. 70–77 (2003) doi: 10.1145/945645.945658
28. Ortony, A., Clore, G. L., Foss, M. A.: The referential structure of the affective lexicon. Cognitive science, vol. 11, no. 3, pp. 341–364 (1987) doi: 10.1016/S0364-0213(87)80010-1

29. Pak, A., Paroubek, P.: Twitter as a corpus for sentiment analysis and opinion mining. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10), pp. 1320–1326 (2010)
30. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: Sentiment classification using machine learning techniques. Proceedings of the ACL-02 conference on Empirical methods in natural language processing, vol. 10, pp. 79–86 (2002) doi: 10.3115/1118693.1118704
31. Read, J.: Recognising affect in text using pointwise-mutual information. Unpublished of Master of Science Dissertation, University of Sussex (2004)
32. Read, J.: Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In: Proceedings of the ACL Student Research Workshop, pp. 43–48 (2005)
33. Rubin, V. L., Stanton, J. M., Liddy, E. D.: Discerning emotions in texts. In: AAAI Conference on Artificial Intelligence (2004)
34. Solorio, T., Liu, Y.: Learning to predict code-switching points. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 973–981 (2008)
35. Solorio, T., Liu, Y.: Part-of-speech tagging for english-spanish code-switched text. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pp. 1051–1060 (2008)
36. Tan, S., Wang, Y., Cheng, X.: Combining learn-based and lexicon-based techniques for sentiment detection without using labeled examples. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, pp. 743–744 (2008) doi: 10.1145/1390334.1390481
37. Turney, P. D., Littman, M. L.: Unsupervised learning of semantic orientation from a hundred-billion-word corpus. Technical Report NRC Technical Report ERB-1094 (2002)
38. Turney, P. D., Littman, M. L.: Measuring praise and criticism: Inference of semantic orientation from association. ACM Transactions on Information Systems (TOIS), vol. 21, no. 4, pp. 315–346 (2003) doi: 10.1145/944012.944013
39. Vyas, Y., Gella, S., Sharma, J., Bali, K., Choudhury, M.: POS tagging of english-hindi code-mixed social media content. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 974–979 (2014) doi: 10.3115/v1/D14-1105
40. Wang, S. I., Manning, C. D.: Baselines and bigrams: Simple, good sentiment and topic classification. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, vol. 2, pp. 90–94 (2012)
41. Wiebe, J.: Learning subjective adjectives from corpora. In: Proceedings of National Conference on Artificial Intelligence, vol. 20 (2000)
42. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. Computational Linguistics, pp. 347–354 (2005)
43. Zhang, L., Ghosh, R., Dekhil, M., Hsu, M., Liu, B.: Combining lexicon-based and learning-based methods for twitter sentiment analysis. HP Laboratories, Technical Report HPL-2011, vol. 89, pp. 1–8 (2011)

ISSN 1870-4069

Topic Modeling Based Analysis of Professors Roles in Directing Theses

Mahdi Mohseni, Heshaam Faili

University of Tehran,
Department of Electrical and Computer Engineering,
Iran

{mahdi.mohseni, hfaili}@ut.ac.ir

Abstract. Topic-sensitive analysis of participants in scientific networks has been a hot research topic for many years. In this paper, we propose a topic modeling-based approach to analyze the influence of professors when they take on different roles (e.g., first supervisor, second supervisor and advisor) in directing theses. We explain how a topic distribution of theses obtained by Latent Dirichlet Allocation provides a basis to formulate the problem and how the unknown parameters are calculated. Results of experiments on a real-world dataset reveal some interesting facts about professor roles in supervising and advising theses. Our results are in contradiction with this traditional view that supervisors are more influential than advisor and first supervisor is a more effective role than second supervisor.

Keywords: Influence analysis, topic modeling, gradient descent.

1 Introduction

Influence analysis of professors in directing theses provides important information which can be used for ranking professors and modifying education regulations in academic institutions. Professors who supervise students in universities have the authority to determine the direction of theses. If supervision is performed by only one professor, he is the only supervisor and has all authority.

But when more than one professor has been involved in a thesis and they take on different roles, e.g., first supervisor, second supervisor and advisor, how can one analyze the influence of each role? One key approach is to make use of topic modeling to achieve topic distribution of theses as a basis for further analyses. Topic modeling has been utilized in this way in many researches.

Studying the problem of identifying influential users of the Twitter network, Weng et al. (2010) utilize topic modeling to identify topics that users of the network are interested in and accordingly a topic-specific network of Twitter users are created.

Then, a method which is an extension of PageRank is applied to find influential users. In (Ramage et al., 2009), topic modeling is applied to PhD dissertation abstracts to find which universities are leading and which are lagging. Underlying topics of the abstracts extracted using Latent Dirichlet Allocation (LDA) (Blei et al., 2003) are

provided to a temporal similarity measure to examine if a university lead or lag other universities.

In this paper, we intend to utilize a standard topic model, namely LDA (Blei et al., 2003), to analyze professor roles in directing theses in a real-world dataset. The dataset consists of graduate theses in the School of Electrical and Computer Engineering at University of Tehran. Based on regulations of this university, a professor can be a supervisor or an advisor in directing a thesis.

Moreover, each thesis can be directed by one, two or, in rare cases, three supervisors and zero, one or, again in rare cases, two advisors. Traditionally, it is supposed that supervisors have more influence on leading a thesis than advisors and similarly first supervisor is a more influential role than second supervisor, and so on. The results of proposed topic modeling-based analysis method contradict this traditional point of view about directing theses.

The innovations of this paper are as follows:

- Proposing a topic modeling-based approach to analyze the influence of professors in directing theses.
- Proposing an optimization function for influence identification of professor roles.
- Determining the influence of professor roles in the School of Electrical and Computer Engineering of Tehran University.

The rest of the paper is organized as follows: in section 2 related works are reviewed; section 3 describes the proposed methodology; experiments and analysis are presented in section 4; and finally, the paper is concluded in section 5.

2 Related Works

Topic models have been successfully applied to identify topics buried in text documents. Since the seminal paper on LDA by Blei et al. (2003), this model has been either a basis for many intuitive approaches or an inspiring method in proposing other topic modeling algorithms to analyze the influence of participants in scientific networks.

TwitterRank (Weng et al., 2010) is an extension of PageRank algorithm to measure the influence of users in Twitter. First, topics that twitterers are interested in are automatically identified according to the tweets they published using LDA model. Then, based on the topics distilled, topic-specific relationship networks among twitterers are constructed. Finally, TwitterRank algorithm is applied to measure the influence taking both the topical similarity between twitterers and the link structure into account.

Using standard topic models in microblogging environments is studied in (Hong et al., 2012). In this paper, a few schemes are proposed to train a topic model on twitter and to compare their quality and effectiveness through experiments on two tasks: predicting popular twitter messages and, classifying twitter users and corresponding messages into topical categories.

In (Hall et al., 2008), topic modeling is applied to ACL Anthology to analyze the development of ideas and trends in the field of computational linguistics over the course of the time. Topic distributions achieved by applying LDA to papers are used to

determine the temporal distribution of topics over years by some post hoc calculations. Ramage et al. (2009) focus on this problem that based on the content of PhD dissertation abstracts, which universities are leading and which are lagging.

To answer this question, underlying topics of PhD dissertation abstracts are extracted using LDA and then by defining a similarity score which is based on topic similarity over years they examine how much a university matches the future or past of other universities. In a similar research, Shi et al. (2010) present an approach to analyze the lead-lag relationships of communities based on topic modeling and temporal information.

Based on this hypothesis that the context in which a cited document appeared in a citing document indicates how the cited authors have influenced the contributions by the citing authors, Kateria et al. (2011) propose two models, author link topic and the author cite topic models to simultaneously model the content of documents, and the interests as well as the influence of authors in certain topics.

Topic-Link LDA (Liu et al., 2009) has been proposed to jointly model the document topics and the social network among authors in one unified model. This model assumes the formation of a link between two documents as a combination of topic similarity and community closeness, which brings both topic modeling and community discovering in one unified model.

Author-Conference-Topic (ACT) (Tang et al., 2008) is a model to simultaneously extract topics of papers, authors, and conferences. Topical Affinity Propagation (TAP) (Tang et al., 2009) has been presented to model topic-level social influence on large networks. Inspiring ACT model (Tang et al., 2008), Kim et al. (2016) propose a model called Author-Journal-Topic (AJT) to take both authors and journals into consideration of topic analysis.

In the next section we will describe how we use topic modeling as the initial step of our approach to determine how professors affect directing graduate theses and how influential the role of each professor is.

3 Methodology

Our approach in analyzing the influence of professors in directing theses consists of several steps which will be explained in detail in this section.

3.1 Topic Modeling

In the first step, the topic distributions of thesis abstracts are needed to be extracted. To do that, LDA which is a standard topic model for extracting underling topics of documents is employed.

In this model, each document is represented by a multinomial distribution of topics and each topic is represented by a multinomial distribution over words. Parameters of the LDA model are α and β, the hyperparameters of Dirichlet distributions, and K, the number of topics. The resulted topic distributions are used for the succeeding computations.

3.2 Research Topics Identification

LDA is an unsupervised topic modeling approach to detect topics in documents. Consequently, all of detected topics in thesis abstracts do not characterize research fields and some of them are stop or common words. Hence, topics are examined manually to remove irrelevant ones. Then, the topic distribution of each document is normalized to make sure that the probability distribution is still valid. Remaining relevant topics each can be assigned a title to describe the corresponding research area that has been followed by researchers.

3.3 Problem Formulation

Education regulations of University of Tehran allow that a thesis is directed by more than one professor. According to that these roles are definable for involved professors: first, second and third supervisor, and first and second advisor. Two roles, third supervisor and second advisor, which rarely occur are ignored. It has been taken for granted that in directing theses supervisors are more influential than advisor and first supervisor is a more effective role than second supervisor. To scrutinize how influential these professor roles are two effective factors should be taken into account: 1) the role of involved professors and, 2) the match between topics of a thesis and the research record of professors.

If $\alpha(prof, thesis)$ is the influence of a professor role, with regard to different definable roles:

$$\alpha(\text{prof}, \text{thesis}) = \begin{cases} 1 & if\ thesis\ has\ only\ one\ supervisor \\ \alpha_1, \alpha_2 & if\ thesis\ has\ one\ supervisor\ and\ one\ advisor \\ \alpha_3, \alpha_4 & if\ thesis\ has\ two\ supervisors \\ \alpha_5, \alpha_6, \alpha_7 & if\ thesis\ has\ two\ supervisors\ and\ one\ advisor \end{cases} . \quad (1)$$

In which the influence of each role is defined for all possible states. The influence of supervisor is 1 if $thesis$ is supervised by only one professor. α_1 and α_2 are respectively the influence of supervisor and advisor when $thesis$ is directed by one supervisor and one advisor ($\alpha_1 + \alpha_2 = 1$).

If $thesis$ is supervised by two supervisors, α_3 and α_4 are the influence of the first and second supervisors ($\alpha_3 + \alpha_4 = 1$). Similarly, in cases that two supervisors and an advisor are involved in $thesis$, α_5, α_6 and α_7 are the influences of them, respectively ($\alpha_5 + \alpha_6 + \alpha_7 = 1$). Based on this definition, as long as $0 \leq \alpha_i \leq 1\ \forall\ i = 1, \dots, 7$ and in each case the influences sum up to 1, these values are valid. In our experiments we initialize $\alpha_i\ \forall\ i = 1, \dots, 7$ with regard to following conditions: $\alpha_1 \geq \alpha_2$, $\alpha_3 \geq \alpha_4$, $\alpha_1 \geq \alpha_2$ and $\alpha_5 \geq \alpha_6 \geq \alpha_7$.

Similarity of a thesis' topics and research interest of supervisor(s) and advisor is another factor in influencing a thesis. The more the topic distribution of a thesis with research records of a professor is matched, the more influence that professor has had in directing that thesis. $p^{year}(topic|prof)$ shows how much a professor has experience in a topic up to a specific year. If $p(topic|thesis)$ is the probability that an arbitrary thesis is about a specific topic, resulted from applying LDA to all theses, the cumulative experience of a professor in that topic in year $year$ is defined as:

$$f^{year}(topic|prof) = \sum_{thesis \in year} \alpha(\text{prof}, \text{thesis}) \times p^{year-1}(topic|prof) \times p(topic|thesis), \quad (2)$$

If $f^{year}(topic|prof)$ is normalized, $p^{year}(topic|prof)$, the experience of a professor in topic $topic$ in that year is obtained:

$$p^{year}(topic|prof) = \frac{f^{year}(topic|prof)}{\sum_{topic'} f^{year}(topic'|prof)}. \quad (3)$$

In formula (2) the unknown parameters, i.e. $\alpha_i \; \forall \, i = 1, \ldots, 7$, are required to be determined.

3.4 Finding Parameters Values

Research interests of professors might change by the passage of the time. However, sharp changes are rarely observed in professors' research directions. They usually get familiar with new areas that they have not worked on and gradually shift to them. This fact is our hypothesis in defining an objective function to determine the unknown parameters of formula (2), the influence of professor role. The objective function is defined as the changes of professors' experiences in consecutive years:

$$F = \sum_{year=2}^{n} \sum_{prof} \sum_{topic} (p^{year}(topic|prof) - p^{year-1}(topic|prof))^2. \quad (4)$$

By minimizing the above objective function with respect to $\alpha_i \; \forall \, i = 1, \ldots, 7$, these unknown parameters can be determined. As this objective function is complex, it is not easy to arrive at an analytical solution to solve it. So, we choose to find the parameters values by gradient descent. In an iterative procedure, the value of each parameter is changed by $\pm\varepsilon$ and F is calculated. In a direction that F is decreased the new value of the parameter is determined which is either raising or falling by ε. This will be repeated until there is no significant change in F value.

4 Experiments and Analysis

4.1 Dataset

The abstracts of graduate theses in the School of Electrical and Computer Engineering of University of Tehran between years 2006-2015 are used in our experiments. The number of theses in these years is 1813.

As it was mentioned before, the rare cases in which a thesis is directed by 3 supervisors or two advisors are removed from the dataset. The number of these theses is 36. The information of the remaining 1717 theses is shown in Table 1.

Table 1. Statistics of the dataset.

Item	Number
All theses	1777
Theses directed by only one supervisor	902
Theses directed by one supervisor and one advisor	542
Theses directed by two supervisor and no advisor	251
Theses directed by two supervisor and one advisor	82

4.2 Experiments

Preprocessing: first of all, the dataset should be normalized, tokenized and lemmatized. Moreover, stop words are needed to be removed. To do that, Persianp Toolbox (Mohseni et al., 2016), which is a free Persian text processing toolbox is used.

Topic Modeling: to obtain the topic distribution of documents and the word distribution of topics, JGibbLDA[1] which is a java implementation of Latent Dirichlet Allocation (LDA) using Gibbs Sampling technique is used. The hyperparameters of Dirichlet distributions, α and β, are both set to 0.01 and the number of topics, K, to 100. The number of iterations is also 1500.

Research Topics Identification: According to 20 top words in the word distribution of each topic, an expert detects which topics consist of stop or common words and so to be removed and which ones are represented research areas and to be assign proper titles. In our experiments, from 100 topics, 79 are detected as research topics and other 21 as irrelevant. In other word, one can claim that research interests of professors in the Department of X at the University of Y can be partitioned into these 79 research areas according to an LDA-based analysis.

Finding Parameters Values: To find the influence of professor roles in directing theses (ref. to sections 3-3 and 3-4), we should initialize $\alpha_i \; \forall \; i = 1, \dots, 7$ (formula (1)) with proper values according to explained conditions. The independent parameters are α_1, α_3, α_5 and α_6 and other parameters are dependent. As assigning roles to professors has not been done recklessly and with no purpose, we initialize the value of each independent parameter in a way that the influence of none of roles is ignored. Moreover, to reach a comprehensive result, the initial value of each parameter is changed in an appropriate range by small rate δ and the experiment is repeated for each new parameter initialization. Finally, the mean and variance of resulted values for each parameter are reported. The change rate of parameter values in computing the gradient descent is set to 0.01. Table 2 summarizes the parameter setting of the problem.

To calculate the cumulative experience of professors (formula (2)), topic distribution of professors, i.e. $p^{year-1}(prof|topic)$, for $year = 1$ is not available. Also, if a professor has had no role in directing theses in the first few years his research record is unknown. To solve these problems, we can initialize the research record of a professor using one of these two strategies: 1) the mean of topic distributions of all theses in all years that a professor has been involved in or, 2) the mean of the topic distribution of theses in the first N years since a professor had been assigned his first role. In our experiments as long as $N \geq 3$ the results of both strategies are the same.

[1] http://jgibblda.sourceforge.net/

Table 2. Parameter setting.

Independent parameters	α_1, α_3, α_5 and α_6
Dependent parameters	α_2, α_4 and α_7
Range of values for parameters	$0.5 \leq \alpha_1 \leq 0.8$ $\alpha_2 = 1 - \alpha_1$ $0.5 \leq \alpha_3 \leq 0.8$ $\alpha_4 = 1 - \alpha_3$ $0.25 \leq \alpha_5 \leq 0.4$ $0.25 \leq \alpha_6 \leq 0.$ $\alpha_7 = 1 - \alpha_5 - \alpha_6$
Change rate to calculate the gradient descent (ε)	0.01
Change rate in initialization of values (δ)	0.05

Table 3. Mean and standard deviation (Stdv) of parameters.

Thesis is directed by	**Role**	**Parameter**	**Mean**	**Stdv**
One supervisor and one advisor	Supervisor	α_1	0.49	0.01
	Advisor	α_2	0.51	0.01
Two supervisors	1st Supervisor	α_3	0.50	0.00
	2nd Supervisor	α_4	0.50	0.00
Two supervisors and one advisor	1st Supervisor	α_5	0.29	0.02
	2nd Supervisor	α_6	0.30	0.01
	Advisor	α_7	0.41	0.02

By examining professors involved in all theses, we recognized that some professors have been involved in only one or few theses. They are professors who have been either new faculty members of Electrical and Computer Engineering Department or from other departments.

Since we don't want that the lack of information about research records of these professors biases our experiments, in optimization procedure (ref. section 3-4) we do not take professors whose name are repeated less than 3 time into account. In this way, new faculty members and professors from outside the department have no bad effect on our calculations.

Based on the aforementioned parameter setting and the initialization of parameters, $\alpha_i \; \forall \; i = 1, \ldots, 7$ is calculated. Table 3 shows the mean and standard deviation of resulted values.

4.3 Discussion

By analyzing the results presented in table 3, some interesting conclusions can be drawn. Above all, low standard deviation of obtained values shows that the proposed approach is robust to initialization of parameters. This is interpretable as dependability of the approach.

In cases that a thesis is directed by two professors, either by one supervisor and one advisor or by two supervisors, their influence are almost the same. This is in contradiction to this traditional point of view that supervisor role has more influence than advisor role and first supervisor is also more influential than second supervisor.

Another interesting conclusion is that when a thesis is supervised by two supervisors and one advisor, advisor role overshadows supervisor roles. This can be justified that when in spite of two supervisors it is still needed that another professor plays a role, it means the thesis covers a topic which is in the expertise of none of supervisors.

5 Conclusions

In this paper professors' roles in supervising or advising graduate theses in the Department of X at the University of Y were studied. Proposing a topic modeling-based approach, we analyzed how professors affect theses and how much influential their roles are when more than one professor has been involved in directing a thesis. For the future work, we aim to use our achievements here in order to analyze the professors' network to study the leading professors and discover the change pattern of research interests in the department.

References

1. Blei, D. M., Ng, A. Y., Jordan, M. I.: Latent Dirichlet allocation. Journal of machine Learning research, vol. 3, pp. 993–1022 (2003)
2. Hall, D., Jurafsky, D., Manning, C. D.: Studying the history of ideas using topic models. In: Proceedings of the conference on empirical methods in natural language processing Association for Computational Linguistics, pp. 363–371 (2008)
3. Hong, L., Davison, B. D.: Empirical study of topic modeling in twitter. In: Proceedings of the first workshop on social media analytics, pp. 80–88 (2010) doi: 10.1145/1964 858.196487
4. Kataria, S., Mitra, P., Caragea, C., Giles, C. L.: Context sensitive topic models for author influence in document networks. Twenty-Second International Joint Conference on Artificial Intelligence, vol. 22, no. 3, pp. 2274 (2011)
5. Kim, H. J., An, J., Jeong, Y. K., Song, M.: Exploring the leading authors and journals in major topics by citation sentences and topic modeling. In: Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries, pp. 42–50 (2016)
6. Liu, Y., Niculescu-Mizil, A., Gryc, W.: Topic-link LDA: Joint models of topic and author community. In: Proceedings of the 26th annual international conference on machine learning, pp. 665–672 (2009) doi: 10.1145/1553374.155346
7. Mohseni, M., Ghofrani, J., Faili, H.: Persianp: A persian text processing toolbox. In: International Conference on Intelligent Text Processing Linguistics, Springer, vol. 9623, pp. 75– 87 (2016) doi: 10.1007/978-3-319-75477-2_4
8. Ramage, D., Manning, C. D., McFarland, D. A.: Which universities lead and lag? Toward university rankings based on scholarly output. In: Proceeding of NIPS Workshop on Computational Social Science and the Wisdom of the Crowds (2010)
9. Shi, X., Nallapati, R., Leskovec, J., McFarland, D., Jurafsky, D.: Who leads whom: Topical lead-lag analysis across corpora. NIPS Workshop (2010)
10. Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su., Z.: Arnetminer: extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 990–998 (2008) doi: 10.1145/ 1401890.14020

11. Tang, J., Sun, J., Wang, C., Yang, Z.: Social influence analysis in large-scale networks. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 807–816 (2009) doi:10.1145/1557019.155710
12. Weng, J., Lim, E. P., Jiang, J., He, Q.: Twitterrank: finding topic-sensitive influential twitterers. In: Proceedings of the third ACM international conference on Web search and data mining, pp. 261–270 (2010) doi: 10.1145/1718487.17185

Electronic edition
Available online: http://www.rcs.cic.ipn.mx

ISSN: 1870-4069
http://rcs.cic.ipn.mx

Centro de Investigación
en Computación