Arabic Textual Entailment with Word Embeddings

Nada Almarwani and **Mona Diab**
Department of Computer Science
The George Washington University
{nadaoh;mtdiab}@gwu.edu

Abstract

Determining the textual entailment between texts is important in many NLP tasks, such as summarization, question answering, and information extraction and retrieval. Various methods have been suggested based on external knowledge sources; however, such resources are not always available in all languages and their acquisition is typically laborious and very costly. Distributional word representations such as word embeddings learned over large corpora have been shown to capture syntactic and semantic word relationships. Such models have contributed to improving the performance of several NLP tasks. In this paper, we address the problem of textual entailment in Arabic. We employ both traditional features and distributional representations. Crucially, we do not depend on any external resources in the process. Our suggested approach yields state of the art performance on a standard data set, ArbTE, achieving an accuracy of 76.2 % compared to current state of the art of 69.3 %.

1 Introduction

Recently, there have been a number of studies addressing the problem of Recognizing Textual Entailment (RTE). The core problem is to recognize semantic variability in textual expression, which can potentially have the same meaning (Dagan et al., 2010). Modeling this phenomenon has a significant impact on various NLP applications, such as question answering, machine translation, and summarization. Textual Entailment (TE) can be defined as a directional entailment relation between a pair of text fragments; if the meaning of the Hypothesis (H) can be inferred from the Text (T) (Dagan et al., 2006). Since the first PASCAL RTE challenge (Dagan et al., 2006) to date, different approaches have been proposed. A popular trend is the use of supervised machine learning approaches that rely on extracting a set of features based on the underlying syntactic/semantic/lexical relation between the TH pair. Most of the approaches have been applied and tested on English TE.

Arabic, on the other hand, has relatively fewer studies for entailment detection. It is one of the most complex languages to process due to its morphological richness and relatively free word order as well as its diglossic nature (where the standard and the dialects mix in most genres of data). Moreover, Arabic still lacks the large scale handcrafted computational resources that have come in very handy for English such as a large WordNet (Miller, 1995) or a resource such as VerbOcean (Chklovski and Pantel, 2004). Hence building a reliable RTE system for Arabic poses more challenges than those faced when dealing with English. Accordingly, in this paper, we propose an approach that does not rely on such external resources but rather on modeling word relations derived from large scale corpora.

The rest of this paper is organized as follows: Section 2 provides an overview of textual entailment works in both English and Arabic, Section 3 describes the basic features and word distributional representation based features, Results and an evaluation of the system are presented in Section 4, and we conclude in Section 5.

2 Related Work

Since the start of the PASCAL RTE challenges in 2005 up until 2011, a large number of methods and approaches have been proposed. The

Proceedings of The Third Arabic Natural Language Processing Workshop (WANLP), pages 185–190,
Valencia, Spain, April 3, 2017.

entailment judgment is typically cast as a classification decision: *true* entailment if the relation holds and *false* otherwise. Therefore, most of the proposed systems have been based on machine learning approaches which model the entailment relation over a variety of conventional features varying from basic lexical features to deep semantic features (Inkpen et al., ; Pakray et al., 2011; Zanzotto and Moschitti, 2006; Malakasiotis and Androutsopoulos, 2007). External semantic resources such as WordNet and VerbOcean have been extensively used to capture the semantic relationships between words in the H and T, and also to further enhance the entailment recognition system (Iftene and Moruz, 2009; Mehdad et al., 2009). Using such resources, the authors explicitly model lexical and semantic features (Zanzotto et al., 2009; Sammons et al., 2009; Clinchant et al., 2006; Mehdad et al., 2009; Wang and Neumann, 2008; Moschitti, 2006). Other methods rely on dependency tree representations using different computations ranging from basic common edge count (Malakasiotis and Androutsopoulos, 2007) to syntactic dependency analysis on corresponding text pairs (Wang and Neumann, 2007).

Recent advances in modeling word representations are shown to be useful for many NLP tasks. Zhao et al., (2015) investigated the effectiveness of word embeddings in different tasks including TE. The focus of this work is Arabic TE, which to the best of our knowledge, has few studies in the entailment literature. In 2011, Alabbas (2011) develops the ArbTE system to assess existing TE techniques when applied to Arabic TE. Later work proposed the use of extended tree edit distance with subtrees resulting in a more flexible matching algorithm to identify TE in Arabic (Alabbas and Ramsay, 2013). Moreover, others have looked closely at negation and polarity as additional features (AL-Khawaldeh, 2015) both of which resulted in better Arabic TE recognition performance, 61% and 69% accuracy, respectively.

3 Approach

Similar to previous approaches to the RTE, we cast the problem as a binary classification task. Namely, we identify if a T entails an H. We model the problem within a supervised framework. We rely on the following set of features in our modeling.

3.1 Features

1. Length: entailment is a directional relation and in general T may include more information, therefore, in most cases T and H are similar in their length or H is shorter than T. Therefore, the following set of features are used to record the length information of a given pair using the following measures:$|B - A|$, $|A \cap B|$, $\frac{(|B|-|A|)}{|A|}$, $\frac{(|A|-|B|)}{|B|}$, $\frac{|A \cap B|}{|B|}$, where $|A|$ represents the number of unique instances in A, $|B - A|$ refers to the number of unique instances that are in B but not in A, and $|A \cap B|$ represents the number of instances that are in both A and B. We applied them at the token, lemma, and stem levels.

2. Similarity score: A similar pair is more likely to share more words and hence the entailment relation holds. Therefore, a two typical similarity measures Jaccard (Jaccard, 1901) and Dice (Dice, 1945) have been used to measure the similarity between the TH pair at the token, lemma, and stem levels. In particular:
$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|}$;
and $Dice(A, B) = \frac{2|A \cap B|}{|A|+|B|}$.

3. Named Entity: Recognizing the similarity and differences between name entity instances in the pair plays an important role in recognizing entailment. Therefore, we use NERAr (Gahbiche-Braham et al., 2014) to extract the following entities: Organization, Person, and Location, then we represent each of them as a bag of words and we use the length based feature explained in 1 resulting in 5 features for each named entity in the extracted categories. For example, if a location name in T appears as "The United State of America" and in H appears as "United States" or "America". Then the length feature $\frac{|A \cap B|}{|B|}$ from 1 gives the percentage of NEs overlapping between T and H; i.e. if "United States" is the NE found, then the percentage overlap between the T and H is 40%, and if the NE is "America" then the percentage overlap is 20%.;

4. Word Embeddings: Word embeddings capture word meaning and paraphrases which should overcome the lack of explicit lexical overlap between the TH pair. We derive word vector representations for about 556K words

using the Word2vec (w2v) (Mikolov et al., 2013) model built using the standard implementation,[1] Namely, we use commonly set parameters: the skip-gram architecture with 300 dimensions for the vectors, window size set to 10. We use inverse document frequency (IDF) scores as dimension values for the input matrix to the w2v. For each TH pair, we obtain the following features: 1. The cosine distance between the T and H vectors which consider both matched and unmatched words; and, b) The cosine distance between the T-H and H-T vectors which consider unmatched words only; specifically, they represent the words in T that are not in H and vice versa, respectively. The latter provides for additional evidence for the distance between T and H. Each vector is calculated as follows:

$$Vector = \sum_{i=1}^{n} IDF(Wi) \cdot w2v(Wi)$$

For example, given the following TH pair:[2]

T: lys bEAlm Algyb AlA Allh.
H: lA ydrk Algyb AlA Allh.
English Translation of Both T and H: Only God knows the unseen.

The T-H vector in the above example is the sum of two vectors: "lA" and "ydrk", which are the words in T and that are not in H, each multiplied by its IDF score.

4 Experiments and Result

4.1 Data

We use the annotated data used in previous studies, ArbTE (Alabbas, 2013), which comprises 600 TH pairs in Modern Standard Arabic (MSA). The ArbTE has been collected from news websites and annotated for entailment manually (Alabbas and Ramsay, 2012). For the word embedding models we use Arabic Gigaword (Parker et al., 2011), the Arabic Treebank (ATB) (Maamouri et al., 2008) and Wikipedia.[3]

All the data, ArbTE and the data used for deriving the word embeddings, are preprocessed in the same manner using the following preprocessing steps: SPLIT (Al-Badrashiny et al., 2016) is used to check if the word is a number, date, URL, or punctuation. Then all URLs and punctuation are removed and numbers and dates are normalized to Num and Date, respectively. Next, Alef and Yaa characters are normalized each to a single form which is typical in large scale Arabic NLP applications. For tokenization, lemmatization and stemming we use MADAMIRA (Pasha et al., 2014). We apply the D3 tokenization scheme which segments determiners as well as proclitics and enclitics; i.e. the D3 tokenization scheme is similar to the ATB scheme with the extra tokenization of the determiner Al. Finally, we remove stop words based on a list,[4] however, we keep negation words as we believe they are important for TE. As a side note, the resulting word vectors cover almost all the words in the ArabTE except for about 30 OOV words most of which are NE and we ignore them during vector calculation.

4.2 Experimental Setup

Our system is a supervised model, therefore, we experiment with multiple supervised frameworks: an SVM classifier (LIBSVM), Logistic Regression (LR) using specifically the LIBLINEAR classifier, and Random Forest (RF). All experiments are implemented using the WEKA software package (Witten and Frank, 2005). All classifiers yield relatively similar performance with the LR classifier obtaining the best results, which is expected since both the feature space and the dataset are relatively small. Therefore, we report results using LR only.

We report results on a development tuning set, DEV, and a TEST set. We devised 3 training protocols: DEV1, DEV5, and DEV10. Given the size of the labeled data, we run our experiments varying the training protocol and tuning steps while keeping the TEST as a held out data set constant for set ups DEV1 and DEV5. The tuning data, DEV1, comprises 10% of the data, our TRAIN1 data corresponds to 80%, and TEST (held out) is 10% of the data. In the second set up, for DEV5 we calculate the average performance as measured across 5-fold cross validation on 90% of the data. In DEV10 we carry out our experiments with 10-fold cross validation on the entire dataset so as to

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

[2] examples are presented using the Buckwalter transliteration system (Buckwalter, 2002)

[3] https://dumps.wikimedia.org/arwiki/20161120/

[4] https://pypi.python.org/pypi/many-stop-words

System	DEV1	DEV10	DEV5	TEST
BOW1	58.75	59.07	59.3	65
BOW2	72.71	72.96	73	65
ETED1 (Alabbas, 2013)		60		
ETED2 (Alabbas, 2013)		65.7		
ETED+ABC (Alabbas, 2013)		67.3		
ATE (AL-Khawaldeh, 2015)		61.7		
SANATE (AL-Khawaldeh, 2015)		69.3		
LR-WE	67.5	64.67	65.56	61.67
LR-TYP	76.66	74.33	74.44	68.33
LR-ALL	**79.16**	**76.2**	**76.48**	**71.67**

Table 1: Performance Accuracy % of our system on ArbTE datasets, along with results yielded by comparative state of the art systems

compare to previous work. We report the results: with typical features (length and similarity score, named entity), specifically, without word embeddings, as **TYP**; with word embeddings features alone, as **WE**; the last setting is with all features combined, as **ALL**, both TYP and WE combined.

We report results on two baselines based on the percentage of common words or lemmas between T and H. BOW1 and BOW2 represent these baselines in Table 1. In BOW1, we represent the overlap score as binary score (0 or 1) according to a predefined threshold of 75% overlap.[5] In the second baseline, BOW2, the word overlap score is used but, different from BOW1, we use the classifier to determine the optimal threshold based on the training data. As can be seen, BOW2 is higher than BOW1 where a threshold is manually set; this is an artifact of the nature of this dataset where there is a high correlation between overlap percentage and the entailment relation, and the cutting point for the entailment learned by the classifier is optimal for this dataset. Therefore, we include both as baselines for the system.

Beside the baselines, Table 1 illustrates the results obtained by previous studies on the same data set. We only have the 10 fold (DEV10) results from other systems. As illustrated, **LR-ALL** condition yields the best results for all test conditions consistently across all data sets improving over the baselines by a significant margin and outperforms **LR-TYP** and **LR-WE**. Other systems (ETED1, ETED2, ETED+ABC, ATE, and SANATE) have approached the Arabic TE in a different way, wherein ETED systems the main focus was on the impact of Tree Edit Distance (TED) on the Arabic TE using different model extension, and in ATE and SANATE systems the focus was on the effect of negation and polarity on the Arabic TE. Our system outperforms these systems significantly. Moreover, **LR-TYP** significantly outperforms **LR-WE** and achieves the best performance among all runs and all three setups.

These results indicate that word embedding based features enhance the accuracy by about 2% increase from the **TYP** based system. The 10 fold cross validation experimental set up is carried out to compare our performance against previous studies, namely, employing the same experimental setups in Alabbas (2013). We can see that our result outperforms other works when using **TYP** and **ALL** which shows that not only the word embedding but also the basic similarity features that have been heavily implemented on the English system have improved the result over the Arabic entailment state of the art, along with explicit NE modeling as a bag of words and the calculation of similarity measures over it. On the other hand, the word embedding based features alone yield comparable results to the other systems.

4.3 Error Analysis and Discussion

In our system, we use the text as a bag of words and ignore their order. Also, we follow a simple basic assumption: that is if the overlap between the TH pair is high then the positive entailment relation holds, and that it fails to hold otherwise. As can be seen from the baseline in Table 1 this assumption works very well in this dataset. When inspecting the dataset, it turns out that the

[5]This is empirically determined in pervious studies in the English RTE system and it has been also used as baseline in the Arabic systems.

Arabic dataset has this as a dominant phenomenon meaning higher overlap induces entailment and vice versa. Furthermore, the word embedding features in our model help in the semantic interpretation of the unmatched words which results in a performance boost of the result as the pair become closer or far apart in the vector space. Thus, the type of errors inspected are for the more complicated pairs. For example, our system failed to detect the lack of entailment relation in the following example:

T: AlElmA' yHtArwn fy ASAbp AlnwE Algryb mn bktryA <y kwlAy Alty Zhrt fy AlmAnyA ntyjp AsthlAk xyAr mn AsbAnyA llnsA' AlbAlgAt Akvr mn gyrhn
T-English-Translation: Scientists are confused about a strange kind of E.coli that has emerged in Germany as the result of the consumption of cucumbers from Spain, which affects adult women more than others.
H: bktryA AlxyAr fy AlmAnyA tSyb AlnsA' Akvr mn AlrjAl
H-English-Translation: Cucumbers bacteria in Germany affects women more than men.

In this example, the H has a specific piece of information which is not in the text, yet our system labels it as a true entailment. Furthermore, NE features in our model are basic features that do not apply any preprocessing or normalization to, for example, map abbreviations, which leads to some errors in our model. In addition, there are different NLP phenomena we did not handle such as co-reference resolution and syntactic parsing, which we believe could improve the performance.

5 Conclusion

This paper shows our work to address the entailment relation in under-resourced languages, specifically Arabic. We have shown that the use of word representation based features provides a reasonable result when compared with other basic surface level matching features. The key characteristic of these features is the fact that they do not depend on external language resources, but are induced in the latent space, namely using a word2vec model that can be easily generated in any language, i.e. an advantage over the required use of external resources. While we have only studied the effect of such features on Arabic, they can easily be applied to other languages. Although the set we evaluated on was limited in size and to types of phenomena that are usually related to entailment, it was sufficient to confirm that, indeed, word embeddings can be used to enhance textual entailment in such languages. Finally, the current system still has limitations including various ways in which word embeddings could be incorporated.

References

Mohamed Al-Badrashiny, Arfath Pasha, Mona Diab, Nizar Habash, Owen Rambow, Wael Salloum, and Ramy Eskander. 2016. Split: Smart preprocessing (quasi) language independent tool. In *10th International Conference on Language Resources and Evaluation (LREC'16)*, Portoro, Slovenia. European Language Resources Association (ELRA).

Fatima T. AL-Khawaldeh. 2015. A study of the effect of resolving negation and sentiment analysis in recognizing text entailment for arabic. *World of Computer Science and Information Technology Journal (WCSIT)*, 5(7):124–128.

Maytham Alabbas and Allan Ramsay. 2012. Dependency tree matching with extended tree edit distance with subtrees for textual entailment. In *FedCSIS*, pages 11–18.

Maytham Alabbas and Allan Ramsay. 2013. Natural language inference for arabic using extended tree edit distance with subtrees. *Journal of Artificial Intelligence Research*, 48:1–22.

Maytham Alabbas. 2011. Arbte: Arabic textual entailment.

Maytham Abualhail Shahed Alabbas. 2013. Textual entailment for modern standard arabic.

Tim Buckwalter. 2002. Arabic transliteration. *URL http://www. qamus. org/transliteration. htm*.

Timothy Chklovski and Patrick Pantel. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In *EMNLP*, volume 4, pages 33–40.

Stéphane Clinchant, Cyril Goutte, and Eric Gaussier. 2006. Lexical entailment for information retrieval. In *Advances in Information Retrieval*, pages 217–228. Springer.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. Recognizing textual entailment: Rational, evaluation and approaches–erratum. *Natural Language Engineering*, 16(01):105–105.

Lee R. Dice. 1945. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

Souhir Gahbiche-Braham, Hélène Bonneau-Maynard, and François Yvon. 2014. Traitement automatique des entités nommées en arabe: détection et traduction. *TAL*, 54(2):101–132.

Adrian Iftene and Mihai-Alex Moruz. 2009. Uaic participation at rte5. *Proceedings of TAC*.

Diana Inkpen, Darren Kipp, and Vivi Nastase. Machine learning experiments for textual entailment.

Paul Jaccard. 1901. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz.

Mohamed Maamouri, Ann Bies, and Seth Kulick. 2008. Enhancing the arabic treebank: a collaborative effort toward new annotation guidelines. In *LREC*.

Prodromos Malakasiotis and Ion Androutsopoulos. 2007. Learning textual entailment using svms and string similarity measures. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 42–47. Association for Computational Linguistics.

Yashar Mehdad, Alessandro Moschitti, and Fabio Massiomo Zanzotto. 2009. Semker: Syntactic/semantic kernels for recognizing textual entailment. In *Proc. of the Text Analysis Conference, Gaithersburg, MD*. Citeseer.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

George A. Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Machine Learning: ECML 2006*, pages 318–329. Springer.

Partha Pakray, Sivaji Bandyopadhyay, and Alexander Gelbukh. 2011. Textual entailment using lexical and syntactic similarity. *International Journal of Artificial Intelligence and Applications*, 2(1):43–58.

Robert Parker, David Graff, Ke Chen, Junbo Kong, and Kazuaki Maeda. 2011. Arabic gigaword fifth edition ldc2011t11. *Philadelphia: Linguistic Data Consortium*.

Arfath Pasha, Mohamed Al-Badrashiny, Mona T. Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In *LREC*, volume 14, pages 1094–1101.

Mark Sammons, V.G. Vinod Vydiswaran, Tim Vieira, Nikhil Johri, Ming-Wei Chang, Dan Goldwasser, Vivek Srikumar, Gourab Kundu, Yuancheng Tu, Kevin Small, et al. 2009. Relation alignment for textual entailment recognition. In *Text Analysis Conference (TAC)*.

Rui Wang and Günter Neumann. 2007. Recognizing textual entailment using a subsequence kernel method.

Rui Wang and Guenter Neumann. 2008. An divide-and-conquer strategy for recognizing textual entailment. In *Proc. of the Text Analysis Conference, Gaithersburg, MD*.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Fabio Massimo Zanzotto and Alessandro Moschitti. 2006. Automatic learning of textual entailments with cross-pair similarities. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 44, page 401.

Fabio Massimo Zanzotto, Marco Pennacchiotti, and Alessandro Moschitti. 2009. A machine learning approach to textual entailment recognition. *Natural Language Engineering*, 15(04):551–582.

Jiang Zhao, Man Lan, Zheng-Yu Niu, and Yue Lu. 2015. Integrating word embeddings and traditional nlp features to measure textual entailment and semantic relatedness of sentence pairs. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE.