Enhancing Automatic Wordnet Construction Using Word Embeddings

Feras Al Tarouti
University of Colorado Colorado Springs
1420 Austin Bluffs Pkwy
Colorado Springs, CO 80918, USA
faltarou@uccs.edu

Jugal Kalita
University of Colorado Colorado Springs
1420 Austin Bluffs Pkwy
Colorado Springs, CO 80918, USA
jkalita@uccs.edu

Abstract

Researchers have shown that a wordnet for a new language, possibly resource-poor, can be constructed automatically by translating wordnets of resource-rich languages. The quality of these constructed wordnets is affected by the quality of the resources used such as dictionaries and translation methods in the construction process. Recent work shows that vector representation of words (word embeddings) can be used to discover related words in text. In this paper, we propose a method that performs such similarity computation using word embeddings to improve the quality of automatically constructed wordnets.

1 Introduction

A wordnet is a lexical ontology of words. High-quality wordnets have been developed for only a few languages. Wordnets, other than the Princeton WordNet (PWN) (Fellbaum, 1998), are typically constructed by one of two approaches. The translation approach translates the PWN to target languages (Saveski and Trajkovski, 2010; Oliver and Climent, 2012; Lam et al., 2014). In contrast, the merge approach builds the semantic taxonomy of a wordnet in a target language, and then aligns it with the Princeton WordNet by generating translations (Gunawan and Saputra, 2010; Rodríguez et al., 2008).

In this paper, we propose a method to enhance the translation approach using word embeddings produced by the *word2vec* algorithm (Mikolov et al., 2013). We produce wordnets in several languages although the current paper focuses only on the new Arabic wordnet we construct.

2 Constructing Initial Wordnet

We start by automatically generating wordnet synsets for a target language T using the method presented by (Lam et al., 2014), which translates synsets from several intermediate wordnets and ranks them. The approach generates wordnet synsets that do not include any semantic links between them. This paper discusses how we construct the semantic links between synsets in T. Figure 1 shows that we take advantage of the fact that the wordnet synsets created in the previous step are aligned with PWN. This means that synsets with the same meaning for different languages share the same synset ID. To construct the links between synsets in our new wordnet TWN for language T, we extract each $synset_i^{TWN}$ from TWN and find the corresponding synset in PWN, $synset_i^{PWN}$. Here, i is the ID of the synset. Then, for each $synset_i^{PWN}$, we extract each semantic relations r$_j$ and all linked $synset_k^{PWN}$ within PWN. Finally, if $synset_k$, i.e., a synset with ID k is present in TWN, we add a link between $synset_i^{TWN}$ and $synset_k^{TWN}$ in the newly constructed TWN.

3 Generating Word Embeddings

In order to validate the synsets we create using translation and obtain relations between them, we use the *word2vec* algorithm (Mikolov et al., 2013) to generate word representations from an existing corpus. The *word2vec* algorithm uses a feedforward neural network to predict the vector representation of

Pair	Cosine Similarity
$(word_1, word_2)$	0.91
$(word_1, word_3)$	0.22
$(word_1, word_4)$	0.82
$(word_2, word_3)$	0.34
$(word_2, word_4)$	0.72
$(word_3, word_4)$	0.12

Table 1: An example of cosine similarity between words in a candidate synset

words within a multi-dimensional language model. $Word2vec$ has two variations: Skip-Gram (SG) and Continuous Bag-Of-Words (CBOW). In the SG version, the neural network predicts words adjacent to a given word on either side, while in the CBOW model the network predicts the word in the middle of a given sequence of words. In the work presented in this paper, we generate representations of words using both models with several different vector and window sizes to obtain the settings for the highest precision. The purpose of the steps discussed next is to improve the quality of synsets produced by the translation process in addition to generating relations among the synsets.

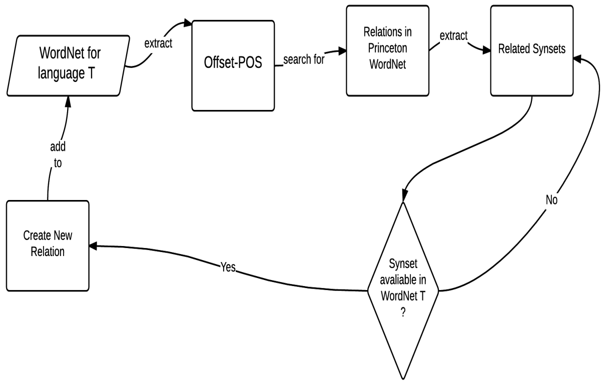

Figure 1: Creating wordnet semantic relations using intermediate wordnet.

3.1 Removing irrelevant words in synsets

We compute the cosine similarity between word vectors within each single synset in TWN, the wordnet being constructed in language T, to filter false word members within synsets. To filter the initially constructed synsets in TWN, we pick a threshold value α such that the selected words have cosine similarity larger than α with each other. For example, let $synset^c_i = \{word_1, word_2, word_3, word_4\}$ be a candidate synset to be potentially included in TWN. We compute the cosine similarity between all the possible pairs of words in $synset^c_i$. Then, we extract the pair of words with the highest cosine similarity. If this pair of words have cosine similarity larger than α, the pair is kept in the final synset $synset_i$, otherwise, $synset^c_i$ itself is discarded. This may have been a low quality candidate synset generated in the translation process. Next, among the remaining words in $synset^c_i$, a word is kept if it has a connection with any word in $synset_i$ with similarity higher than α. For example, let us assume that the cosine similarity between the words in $synset^c_i$ are as shown in Table 1 and α=0.70. First, the pair with the highest cosine similarity, $(word_1, word_2)$ is kept in the final $synset_i$ since its cosine similarity is larger than α. Then, $word_3$ is discarded since it does not have any cosine similarity larger than α with any of the words in the current final $synset_i$. Finally, $word_4$ is kept $synset_i$ since it does have a cosine similarity with $word_1$ that satisfies the threshold α.

3.2 Validating candidate relations

Similarly, we compute the cosine similarity between words within pairs of semantically related synsets. This allow us to verify the constructed relations between synsets in TWN. For example, let $synset_i = \{word_{i1}, word_{i2}, word_{i3}, word_{i4}\}$, $synset_j = \{word_{j1}, word_{j2}, word_{j3}, word_{j4}\}$ be synsets in TWN. And let ρ_{ij} be a candidate semantic relation between $synset_i$ and $synset_j$. We compute the cosine similarity between all the possible pairs of words from $synset_i$ to $synset_j$ and obtain the maximum similarity obtained. Then, if this value is larger than a threshold α_ρ, then we retain the relation ρ_{ij}, otherwise, we discard it.

3.3 Selecting thresholds

To pick the synset similarity threshold value α and the threshold α_ρ for each semantic relation we create, we compute the cosine similarity between pairs of synonym words, semantically related words, and non-related words obtained from existing wordnets. Then, based on the previous data, we select the threshold values that are associated with higher precision and maximum coverage.

4 Experiments

We discuss the generation of a wordnet for Arabic as an example although we have worked with several other languages.

4.1 Datasets and Resources Used

To construct the core wordnet, i.e., wordnet synsets, we use the Microsoft Translator to translate English synsets from PWN to Arabic synsets. We have selected the Microsoft Translator because it gives acceptable quality free of cost. For generating vector representations of the Arabic Words we use the following freely available corpora: Watan-2004 corpus (12 million words) (Abbas et al., 2011), Khaleej-2004 corpus (3 million) (Abbas and Smaili, 2005) and 21 million words of Wikipedia[1] Arabic articles, combined to a single file.

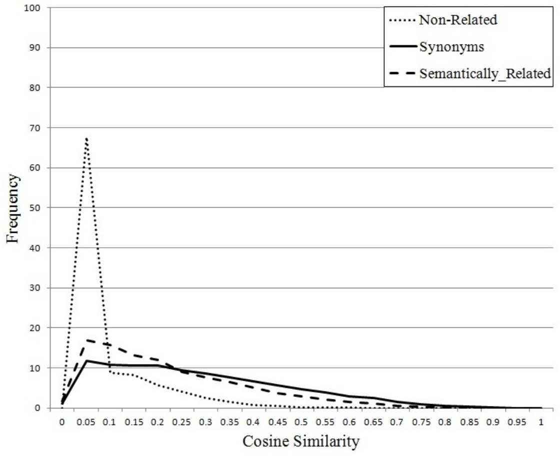

Figure 2: A histogram of synonyms, semantically related words, and non-related words extracted from AWN.

In order to compute the synset similarity threshold value α and the threshold for each semantic relation α_ρ, we use the freely available Arabic wordnet (AWN) (Rodríguez et al., 2008). AWN was manually constructed in 2006 and has been semi-automatically enhanced and extended several times. We start by extracting synonym words, semantically related words, and non-related words from AWN. Then, we use the histogram representation of the cosine similarity of the previous sets of words to set the thresholds. As Figure 2 shows, more than 67% of the non-related words have cosine similarity less than 0.1, while about 23% of the synonym words in AWN have a cosine similarity less than 0.1. Furthermore, about 34% of the semantically related words in AWN have cosine similarity less than 0.1. Table 2 shows the weighted average cosine similarity between synonyms, hypernyms, topic-domain related, part-holonyms, instance-hypernyms, and member-meronyms in AWN where the frequency of the similarity value is the weight.

Relation	Weighted Average Similarity
Synonyms	0.28
Hypernyms	0.22
TopicDomains	0.23
PartHolonyms	0.28
InstanceHypernyms	0.08
MemberMeronyms	0.29

Table 2: The weighted average similarity between related words in AWN.

Relation	Count
Hypernyms	$55,336$
MemberMeronyms	$11,408$
SimilarTo	$10,826$
PartHolonyms	$7,051$
TopicDomains	$3,863$

Table 3: Number of Arabic synset relations we create.

4.2 Creating an Arabic Wordnet

We choose the algorithm for creating wordnet synsets presented by (Lam et al., 2014) because it requires a limited number of freely available resources, which makes it applicable to resource-poor languages. Then, we apply the method we propose in Section 2 to create semantic links between Arabic synsets. Table 3 shows statistics of some of the created links between synsets in our Arabic wordnet.

4.3 Producing Word Embeddings for Arabic

We test the $word2vec$ algorithm with different window sizes. We generate word embeddings using the CBOW version with window sizes 3, 5 and 8. Next, we compute the weighted averages of the cosine similarity between the synonyms in AWN. The highest weighted average we obtained was 0.288 with window size 3, while the weighted averages obtained with window sizes 5 and 8 were 0.283 and

[1]https://ar.wikipedia.org

Algorithm	**Vector Size**	**Similarity Average**
SG	100	0.289
SG	200	0.258
SG	500	0.194
CBOW	100	0.288
CBOW	200	0.259
CBOW	500	0.195

Table 4: Comparison between the weighted similarity average obtained using different $word2vec$ settings.

Threshold	**AWN**	**Our Arabic WordNet**
0.000	$5,941$	$17,349$
0.100	$3,433$	$2,073$
0.288	$2,471$	943
0.500	$1,190$	271
0.750	209	13

Table 5: Comparison between the number of synsets in AWN and our Arabic wordnet using different threshold values.

0.277 respectively. Then, we compare between the SG and the CBOW with different vector sizes. Table 4 shows the weighted average cosine similarity obtained between $16,000$ pairs of synonyms in AWN using both variations of $word2vec$, with window size=3 and vector size set to 100, 200, and 500. We notice that both versions produce almost similar results with a slight advantage to SG with the cost of more execution time. However, for the corpus we use, smaller vector size produces better precision.

4.4 Evaluation & Discussion

We compute cosine similarity between semantically related words extracted from our initial Arabic wordnet produced in Section 4.2. The language model to calculate the cosine similarity is created using CBOW with vector size=100 and window size=3. Table 5 shows a comparison between the number of Arabic synsets we create and the number of synsets in AWN.

We notice that the translation method we use produces high number of synsets compared to the manually constructed AWN. However, the number of synsets sharply decreases after filtering the initial synonyms using the method described in Section 3. Although our Arabic wordnet is automatically created, the number of synsets we create is 60% of the number of synsets in the manually created AWN when filtering the synsets using α= 0.1.

	Threshold Range		
	0- 0.1	0.1 - 0.288	0.288 - 1
Synonyms	34.8%	56.8%	78.4%
Hypernyms	45.2%	57.2%	84.4%
PartHolonym	50.8%	75.2%	90.4%
Member-Meronym	40.8%	56.8%	79.6%

Table 6: Precision of the Arabic wordnet we create.

We evaluate precision by comparing 600 pairs of synonyms, hypernyms, part-holonyms, and member-meronyms with three ranges of cosine similarity values: 0 to 0.1, 0.1 to 0.288, and 0.288 to 1. We asked 3 Arabic speakers to evaluate the pairs using a 0 to 5 scale where 0 represents the minimum score and 5 represents the maximum score. We compute precision by taking the average score and converting it to a percentage. See Table 6.

The precision of the synonyms, hypernyms, part-holonyms, and member-meronyms we produce is 78.4%, 84.4%, 90.4%, and 79.6% respectively, with the threshold set to 0.288. This is higher than the precision obtained by (Lam et al., 2014) which produces synonyms with 76.4% precision when just using PWN. Our results suggest that using lower precision for producing synsets reduces the quality of the other created semantic relations. Our results clearly show that pairs with higher cosine similarity are more likely to be semantically related. It confirms the benefit of combining the translation method with word embeddings in the process of automatically generating new wordnets.

5 Conclusion & Future Work

In this paper, we discuss an approach for automatically generating new wordnets for low-resource languages. Our approach takes advantage of word embeddings to enhance the translation method for automatic wordnet creation. We present an application of our approach to producing new Arabic WordNet. Our method automatically produces Arabic synonyms with 78.4% precision and semantically related pairs of words with up to 90.4% precision. Currently, we are in the process of applying our method to other languages such as Assamese, Bengali and Tamil.

References

Mourad Abbas and Kamel Smaili. 2005. Comparison of topic identification methods for arabic language. In *International Conference on Recent Advances in Natural Language Processing-RANLP 2005*, volume 14.

Mourad Abbas, Kamel Smaïli, and Daoud Berkani. 2011. Evaluation of topic identification methods on arabic corpora. *JDIM*, 9(5):185–192.

Christiane Fellbaum. 1998. *Wordnet: An electronic lexical database*. MIT Press Cambridge.

G Gunawan and Andy Saputra. 2010. Building synsets for Indonesian Wordnet with monolingual lexical resources. In *Asian Language Processing (IALP), 2010 International Conference on*, pages 297–300. IEEE.

Khang Nhut Lam, Feras A. Tarouti, and Jugal Kalita. 2014. Automatically constructing wordnet synsets. In *52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014), Baltimore, USA, June*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751.

Antoni Oliver and Salvador Climent. 2012. Parallel corpora for wordnet construction: Machine translation vs. automatic sense tagging. In *Computational Linguistics and Intelligent Text Processing*, pages 110–121. Springer.

Horacio Rodríguez, David Farwell, Javi Ferreres, Manuel Bertran, Musa Alkhalifa, and Maria Antònia Martí. 2008. Arabic wordnet: Semi-automatic extensions using bayesian inference. In *LREC*.

Martin Saveski and Igor Trajkovski. 2010. Automatic construction of wordnets by using machine translation and language modeling. In *13th Multiconference Information Society, Ljubljana, Slovenia*.