Semantic Similarity of Arabic Sentences with Word Embeddings
El Moatez Billah Nagoudi, Didier Schwab

To cite this version:
El Moatez Billah Nagoudi, Didier Schwab. Semantic Similarity of Arabic Sentences with Word Embeddings. Third Arabic Natural Language Processing Workshop, Apr 2017, Valencia, France. pp.18 - 24. hal-01683485

HAL Id: hal-01683485
https://hal.archives-ouvertes.fr/hal-01683485
Submitted on 15 Jan 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers. L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Abstract

Semantic textual similarity is the basis of countless applications and plays an important role in diverse areas, such as information retrieval, plagiarism detection, information extraction and machine translation. This article proposes an innovative word embedding-based system devoted to calculate the semantic similarity in Arabic sentences. The main idea is to exploit vectors as word representations in a multidimensional space in order to capture the semantic and syntactic properties of words. IDF weighting and Part-of-Speech tagging are applied on the examined sentences to support the identification of words that are highly descriptive in each sentence. The performance of our proposed system is confirmed through the Pearson correlation between our assigned semantic similarity scores and human judgments.

Keywords: Semantic Sentences Similarity, Word Embedding, Word Representations, Space Vector Model.

1 Introduction

Text Similarity is an important task in several application fields, such as information retrieval, plagiarism detection, machine translation, topic detection, text classification, text summarization and others. Finding similarity between two texts, paragraphs or sentences, is based on measuring, directly or indirectly, the similarity between words.

There are two known types of words similarity: lexical and semantic. The first one handles the words as a stream of characters: words are similar lexically if they share the same characters in the same order (Manning et al., 2008). There are many techniques of lexical similarity measures, the most known are: Damerau-Levenshtein (Levenshtein, 1966), Needleman Wunsch (Needleman and Wunsch, 1970), LCS (Chvatal and Sankoff, 1975), JaroWinkler (Winkler, 1999), etc.

The second type aims to quantify the degree to which two words are semantically related. As an example they can be, synonyms, represent the same thing or they are used in the same context. The classical way to measure this semantic similarity is by using linguistic resources, like WordNet (Miller, 1995), HowNet (Dong and Dong, 2003), BabelNet (Navigli and Ponzetto, 2012) or Dbinary (Sérasset, 2015). However, the word embedding techniques can be a more effective alternative to these linguistic databases (Mikolov et al., 2013a).

In this article we focus our investigation on measuring the semantic similarity between short Arabic sentences using word embedding representations. We also consider the IDF weighting and Part-of-Speech tagging techniques in order to improve the identification of words that are highly descriptive in each sentence.

The rest of this article is organized as follows, the next section describes work related to word representations in vector space. In Section 3, we present three variants of our proposed word embedding-based system. Section 4 describes the experimental results of this study. Finally, our conclusion and some future research directions are drawn in Section 5.

2 Word Embedding Models

Words representations as vectors in a multidimensional space allows to capture the semantic and syntactic properties of the language (Mikolov et al., 2013a). These representations can serve as a fundamental building unit to many applications of
Natural Language Processing (NLP). In the literature, several techniques are proposed to build vectorized space representations.

For instance, Collobert and Weston (2008) have proposed a unified system based on a deep neural network architecture, and trained jointly with many well known NLP tasks, including: Chunking, Part of Speech tagging, Named Entity Recognition and Semantic Role Labeling. Their word embedding model is stored in a matrix $M \in \mathbb{R}^{d \times |D|}$, where $D$ is a dictionary of all unique words in the training data, and each word is embedded into a d-dimensional vector. The sentences are represented using the embeddings of their forming words. A similar idea was independently proposed and used by Turian et al. (Turian et al., 2010).

Mnih and Hinton (2009) have proposed another form to represent words in vector space, named Hierarchical Log-Bilinear Model (HLBL). Like virtually all neural language models, the HLBL model represents each word with a real-valued feature vector. For n-gram word-based, HLBL concatenates the $n-1$ first embedding words $(w_1, \ldots, w_{n-1})$ and learns a neural linear model to predicate the last word $w_n$. 

Mikolov et al. (Mikolov et al., 2013c) have used a recurrent neural network (RNN) (Mikolov et al., 2010) to build a neural language model. The RNN encode the context word by word and predict the next word. The weights of the trained network are used as the words embeddings vectors. 

Mikolov et al. (Mikolov et al., 2013a) (Mikolov et al., 2013b) have proposed two other approaches to build a words representations in vector space. using a simplified version of Bengio et al. (Bengio et al., 2003) neural language mode. They replaced the hidden layer by a simple projection layer in order to boost performance. In their work, two models are presented: the continuous bag-of-words model (CBOW) (Mikolov et al., 2013a), and the skip-gram model (SKIP-G) (Mikolov et al., 2013b).

In the first one, the continuous bag of word model CBOW (Mikolov et al., 2013a), predicts a pivot word according to the context by using a window of contextual words around it. Given a sequence of words $S = w_1, w_2, \ldots, w_i$, the CBOW model learns to predict all words $w_i$ from their surrounding words $(w_{k-1}, \ldots, w_{k-1}, w_{k+1}, \ldots, w_{k+1})$. The second model SKIP-G, predicts surrounding words of the current pivot word $w_k$ (Mikolov et al., 2013b).

Pennington et al. (Pennington et al., 2014) proposed a Global Vectors (GloVe) to build a words representations model. GloVe uses the global statistics of word-word co-occurrence to build co-occurrence matrix $M$. Then, $M$ is used to calculate the probability of word $w_i$ to appear in the context of another word $w_j$, this probability $P(i/j)$ represents the relationship between words.

3 System Description

3.1 Model Used

In (Mikolov et al., 2013a), all the methods (Collobert and Weston, 2008), (Turian et al., 2010), (Mnih and Hinton, 2009), (Mikolov et al., 2013c) have been evaluated and compared, and they show that CBOW and SKIP-G are significantly faster to train with better accuracy compared to these techniques. For this reason, we have used the CBOW word representations for Arabic model\(^1\) proposed by Zahran et al. (Zahran et al., 2015). To train this model, they have used a large collection from different sources counting more than 5.8 billion words:

- Arabic Wikipedia (WikiAr, 2006).
- BBC and CNN Arabic corpus (Saad and Ashour, 2010).
- The open parallel corpus (Tiedemann, 2010).
- Arabase Corpus (Raafat et al., 2013).
- Osac: Open source arabic corpora. (Saad and Ashour, 2010).
- MultiUN corpus (Chen and Eisele, 2012).
- AGC Arabic Gigaword Corpus.
- King Saud University corpus (ksucorpus, 2012).
- Meedan Arabic corpus (Meedan, 2012).
- LDC Arabic newswire.
- Raw Quran text (Quran, 2007).
- KDE4 localization files (Tiedemann, 2009).
- Khaleej and Watan 2004 (Khaleej, 2004).

Training the Arabic CBOW model require choice of some parameters affecting the resulting vectors. All the parameters used by Zahran et al. (Zahran et al., 2015) are shown in Table 1.

\(^1\)https://sites.google.com/site/mohazahran/data
The Arabic CBOW Model Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector size</td>
<td>300</td>
</tr>
<tr>
<td>Window</td>
<td>5</td>
</tr>
<tr>
<td>Sample</td>
<td>1e−5</td>
</tr>
<tr>
<td>Hierarchical Softmax</td>
<td>NO</td>
</tr>
<tr>
<td>Negative</td>
<td>10</td>
</tr>
<tr>
<td>Freq. thresh.</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 1: Training configuration parameters

Where:

- **Vector size**: dimensionality of the word vectors.
- **Window**: number of words considered around the pivot word (context).
- **Sample**: threshold for sub-sampling of frequent words.
- **Hierarchical Softmax**: approximation of the full softmax used to predict words during training.
- **Negative**: number of negative examples in the training.
- **Frequency threshold**: threshold to discard less frequent words.

### 3.2 Words Similarity

We used CBOW model in order to identify the near matches between two words \( w_i \) and \( w_j \) (e.g. synonyms, singular, plural, feminization or closely related semantically). The similarity between \( w_i \) and \( w_j \) is obtained by comparing their vector representations \( v_i \) and \( v_j \) respectively. The similarity between \( v_i \) and \( v_j \) can be evaluated using the cosine similarity, euclidean distance, Manhattan distance or any other similarity measure functions. For example, let “الجامعة" (university), “المست '" (evening) and “الكلية" (faculty) be three words. The similarity between them is measured by computing the cosine similarity between their vectors as follows:

\[
\text{sim}(\text{الجامعة}, \text{المست}) = \cos(V(\text{المست}), V(\text{الجامعة})) = 0.13
\]

\[
\text{sim}(\text{الكلية}, \text{الجامعة}) = \cos(V(\text{الكلية}), V(\text{الجامعة})) = 0.72
\]

That means that, the words “كلية" (faculty) and “الجامعة" (university) are semantically closer than “المست" (evening) and “الجامعة" (university).

### 3.3 Sentences similarity

Let \( S_1 = w_1, w_2, ..., w_i \) and \( S_2 = w'_1, w'_2, ..., w'_j \) be two sentences, their word vectors are \((v_1, v_2, ..., v_i)\) and \((v'_1, v'_2, ..., v'_j)\) respectively. We have used three methods to measure the similarity between sentences. Figure 1 illustrates an overview of the procedure for computing the similarity between two candidate sentences in our system.

![Figure 1: The architecture of the proposed system](image)

In the following, we explain our proposed methods to compute the semantic similarity among sentences.

#### 3.3.1 No Weighting Method

A simple way to compare two sentences, is to sum their words vectors. In addition, this method can be applied to any size of sentences. The similarity between \( S_1 \) and \( S_2 \) is obtained by calculating the cosine similarity between \( V_1 \) and \( V_2 \), where:

\[
\begin{align*}
V_1 &= \sum_{k=1}^{i} v_k \\
V_2 &= \sum_{k=1}^{j} v'_k
\end{align*}
\]

For example, let \( S_1 \) and \( S_2 \) be two sentences:

\( S_1 = "\text{ذهب يوسف إلى الكلية}" (Joseph went to college). \)

\( S_2 = "\text{يوفس يذهب مسرعا للجامعة}" (Joseph goes quickly to university). \)

The similarity between \( S_1 \) and \( S_2 \) is obtained as follows:

**step 1: Sum of the word vectors**
V_1 = V(ذهب) + V(الكلية) + V( يوسف) + V(英文) + V( الجامعة)
V_2 = V(الكلية) + V(英文) + V( الجامعة)

step 2: Calculate the similarity

The similarity between S_1 and S_2 is obtained by calculating the cosine similarity between V_1 and V_2.

\[
sim(S_1, S_2) = \cos(V_1, V_2) = 0.71
\]

In order to improve the similarity results, we have used two weighting functions based on the Inverse Document Frequency (Salton and Buckley, 1988) and the Part-Of-Speech tagging (POS tagging) (Schwab, 2005) (Lioma and Blanco, 2009).

3.3.2 IDF Weighting Method

In this variant, the Inverse Document Frequency IDF concept is used to produce a composite weight for each word in each sentence. The IDF weighting of words (Salton and Buckley, 1988) is traditionally used in information retrieval (Turney and Pantel, 2010) and can be employed in our system. The \textit{idf weight} serves as a measure of how much information the word provides, that is, whether the term that occurs infrequently is good for discriminating between documents (in our case sentences).

This technique uses a large collection of document (background corpus), generally the same genre as the input corpus that is to be semantically verified. In order to compute the \textit{idf weight} for each word, we have used the BBC and CNN Arabic corpus\(^2\) (Saad and Ashour, 2010) as a background corpus. In fact, the \textit{idf} of each word is determined by using the formula:

\[
idf(w) = \log\left(\frac{S}{WS}\right)
\]

where \(S\) is the total number of sentences in the corpus and \(WS\) is the number of sentences containing the word \(w\). The similarity between \(S_1\) and \(S_2\) is obtained by calculating the cosine similarity between \(V_1\) and \(V_2\), \(\cos(V_1, V_2)\) where:

\[
\begin{align*}
V_1 &= \sum_{k=1}^{i} \text{idf}(w_k) \times v_k \\
V_2 &= \sum_{k=1}^{i} \text{idf}(w'_k) \times v'_k
\end{align*}
\]

and \text{idf}(w_k) is the weight of the word \(w_k\) in the background corpus.

\(^2\)https://sourceforge.net/projects/ar-text-mining/files/Arabic-Corpora/

Example: let’s continue with the sentences of the previous example, and suppose that IDF weights of their words are:

<table>
<thead>
<tr>
<th>الجامعه</th>
<th>يوسف</th>
<th>الكلية</th>
<th>ادعاء</th>
<th>ذهب</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.27</td>
<td>0.37</td>
<td>0.31</td>
<td>0.29</td>
<td>0.22</td>
</tr>
</tbody>
</table>

step 1: Sum of vectors with IDF weights

\[
\begin{align*}
V_1 &= V(ذهب) \times 0.31 + V(الكلية) \times 0.37 + V( يوسف) \times 0.27 \\
V_2 &= V(الكلية) \times 0.34 + V(英文) \times 0.22 + V( الجامعة) \times 0.29 \\
&+ V( يوسف) \times 0.37
\end{align*}
\]

step 2: Calculate the similarity

The cosine similarity is applied to compute a similarity score between \(V_1\) and \(V_2\).

\[
sim(S_1, S_2) = \cos(V_1, V_2) = 0.78
\]

We note that the similarity result between the two sentences is better than the previous method.

3.3.3 Part-of-Speech weighting Method

An alternative technique is the application of the Part-Of-Speech tagging (POS tag) for identification of words that are highly descriptive in each input sentence (Schwab, 2005) (Lioma and Blanco, 2009). For this purpose, we have used the POS tagger for Arabic language proposed by G. Brah- am et al. (Gahbiche-Braham et al., 2012) to estimate the part-of-speech of each word in sentence. Then, a weight is assigned for each type of tag in the sentence. For example, \(verb = 0.4\), \(noun = 0.5\), \(adjective = 0.3\), \(preposition = 0.1\), etc.

The similarity between \(S_1\) and \(S_2\) is obtained in three steps (Schwab, 2005) as follows:

step 1: POS tagging

In this step the POS tagger of G. Braham et al. (Gahbiche-Braham et al., 2012) is used to estimate the POS of each word in sentence.

\[
\begin{align*}
\text{Pos}_\text{tag}(S_1) &= \text{Pos}_{w_1}, \text{Pos}_{w_2}, ..., \text{Pos}_{w_i} \\
\text{Pos}_\text{tag}(S_2) &= \text{Pos'}_{w'_1}, \text{Pos'}_{w'_2}, ..., \text{Pos'}_{w'_j}
\end{align*}
\]

The function \(\text{Pos}_\text{tag}(S_i)\) returns for each word \(w_k\) in \(S_i\) its estimated part of speech \(\text{Pos}_{w_k}\).

step 2: POS weighting

At this point we should mention that, the weight of each part of speech can be fixed empirically. Indeed, we based on the training data of SemEval-
2017 (Task 1)\(^3\) to fix the POS weights.

\[
\begin{align*}
V_1 &= \sum_{k=1}^i \text{Pos.weight}(\text{Pos}w_k) \cdot v_k \\
V_2 &= \sum_{k=1}^j \text{Pos.weight}(\text{Pos}w'_k) \cdot v'_k
\end{align*}
\]

where \(\text{Pos.weight}(\text{Pos}w_k)\) is the function which return the weight of POS tagging of \(w_k\).

**step 3: Calculate the similarity**

Finally, the similarity between \(S_1\) and \(S_2\) is obtained by calculating the cosine similarity between \(V_1\) and \(V_2\) as follows:

\[
sim(S_1, S_2) = \cos(V_1, V_2)
\]

**Example:** let us continue with the same example above.
- \(S_1 = "\)ذهّب يوسف إلى الكلية" (Joseph went to college).
- \(S_2 = "\)يوفس يعتزم مسرعاً للجامعة" (Joseph goes quickly to university).

and suppose that POS weights are:

<table>
<thead>
<tr>
<th>verb</th>
<th>noun</th>
<th>noun_prop</th>
<th>adj</th>
<th>prep</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>0.5</td>
<td>0.7</td>
<td>0.3</td>
<td>0.1</td>
</tr>
</tbody>
</table>

**step 1: Pos tagging**

The function \(\text{Pos.tag}(S_i)\) is applied to each sentence.

\[
\begin{align*}
\text{Pos.tag}(S_1) &= \text{verb} \text{ noun} \text{ noun_prop} \text{ noun} \\
\text{Pos.tag}(S_2) &= \text{noun_prop} \text{ verb} \text{ adj} \text{ noun}
\end{align*}
\]

**step 2: Sum of vectors with POS weighting**

\[
V_1 = V(\text{كلية}) \cdot 0.5 + V(\text{يوفس}) \cdot 0.7 + V(\text{ذهب}) \cdot 0.4 + V(\text{يرجى}) \cdot 0.4 \\
V_2 = V(\text{مشرعاً}) \cdot 0.5 + V(\text{يوفس}) \cdot 0.3 + V(\text{على}) \cdot 0.4 + V(\text{يوفس}) \cdot 0.7
\]

**step 3: Calculate the similarity**

\[
sim(S_1, S_2) = \cos(V_1, V_2) = 0.82
\]

4 Experiments And Results

4.1 Test Sample

In order to measure effectively the performances of our system, a large collection are necessary. In fact, we have used a dataset of 750 pairs of sentences drawn from publicly Microsoft Research Video Description Corpus (MSR-Video) (MSR-video, 2016), and manually translated into Arabic. The sentence pairs have been manually tagged by four annotators, and the similarity score is the mean of the annotators. This score is a float number between "0" (indicating that the meaning of sentences are completely independent) to "1" (signifying meaning equivalence).

4.2 Preprocessing

In order to normalize the sentences for the semantic similarity step, a set of preprocessing are performed on the data set. All sentences went through by the following steps:

1. Stop-word removal.
2. Remove punctuation marks, diacritics and non letters.
3. We normalized \(\$\) to \(\$\) and \(\&\) to \(\&\).
4. Replace final \(\) followed by \(\) with \(\).
5. Normalizing numerical digits to the token "Num".

4.3 Results

To evaluate the performance of our system, our three approaches were assessed based on their accuracy on the 750 sentences in the MSR-Video corpus. An example of our results is shown in Table 2.

![Table 2: Example of sentence similarity results](http://alt.qcri.org/semeval2017/task1/data/uploads/)

\(\)http://alt.qcri.org/semeval2017/task1/data/uploads/
ever, the similarity score is not good enough when two sentences share the same words, but with a totally different meaning, like in the last pair of sentences.

On the other hand, we calculate the Pearson correlation between our assigned semantic similarity scores and human judgements. The results are presented in Table 3.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic method</td>
<td>72.33%</td>
</tr>
<tr>
<td>IDF-weighting method</td>
<td>78.20%</td>
</tr>
<tr>
<td>POS tagging method</td>
<td>79.69%</td>
</tr>
</tbody>
</table>

Table 3: Correlation results

These results indicate that when the no weighting method is used the correlation rate reached 72.33%. Both IDF-weighting and POS tagging approaches significantly outperformed the correlation to more than 78% (respectively 78.2% and 79.69%).

5 Conclusion and Future Work

In this article, we presented an innovative word embedding-based system to measure semantic relations between Arabic sentences. This system is based on the semantic properties of words included in the word-embedding model. In order to make further progress in the analysis of the semantic sentence similarity, this article showed how the IDF weighting and Part-of-Speech tagging are used to support the identification of words that are highly descriptive in each sentence. In the experiments we have shown how these techniques improve the correlation results. The performance of our proposed system was confirmed through the Pearson correlation between our assigned semantic similarity scores and human judgements. As future work, we can make more improvement in the semantic similarity results by a smart hybridisation between both IDF weighting and POS tagging techniques.

References


