A Bi-recursive Auto-encoders for Learning Semantic Word Embedding

Amal Bouraoui, Salma Jamoussi and Abdelmajid Ben Hamadou
Multimedia InfoRmation systems and Advanced Computing Laboratory MIRACL, Sfax University, Technopole of Sfax, Av.Tunis Km 10 B.P. 242, Sfax, 3021, Tunisia

Keywords: Deep Learning, Word Embedding, Word Semantic, Recursive Auto-encoders.

Abstract: The meaning of a word depends heavily on the context in which it is embedded. Deep neural network have recorded recently a great success in representing the words' meaning. Among them, auto-encoders based models have proven their robustness in representing the internal structure of several data. Thus, in this paper, we present a novel deep model to represent words meanings using auto-encoders and considering the left/right contexts around the word of interest. Our proposal, referred to as Bi-Recursive Auto-Encoders (Bi-RAE), consists in modeling the meaning of a word as an evolved vector and learning its semantic features over its set of contexts.

1 INTRODUCTION

Building good representation of the word meaning is a crucial step in developing well-performing methods for text understanding and processing. Vector space models have recently emerged as powerful technique of representing word semantics. In particular, distributed word representation approach, inspired by various neural architectures, has become the most widely used. Consequently, various methods have been proposed to embed words in lower-dimensional dense real-valued vector space. An intuitive idea is to encode one word into a single vector containing the semantic information of the word in a corpus. Word2Vec model proposed in(Mikolov et al., 2013), using the continuous bag-of-words (CBOW) or the Skip-gram, is one of the famous embedding models. The CBOW architecture is used to predict a word considering its surrounding words, while the Skip-gram architecture is employed to predict the surrounding words of a given word. However, encountering a word in diverse contexts poses a major challenge for machines to understand its meaning. To address this issue, some researchers proposed to learn multiple representations per individual word representing its individual meanings (Neelakantan et al., 2015; Huang et al., 2012). Strategies often focus on discrimination using semantic information extracted from knowledge sources such as WordNet or using clustering algorithms. TF-IDF (Reisinger and Mooney, 2010), (Huang et al., 2012) model and MSSG (Neelakantan et al., 2015) used cluster-based techniques to cluster the context of a word and comprehend word senses from the cluster centroids. (Fei et al., 2014) suggested the use of EM-based probabilistic clustering to assign word senses. A slightly different approach was presented in (Liu et al., 2015) where the authors employed topic modeling to discover multiple word senses. (Nguyen et al., 2017) suggested an extension of the (Liu et al., 2015) model. Researchers argued that multiple senses might be triggered for a word in a given context and replaced the selection of the most suitable sense in (Liu et al., 2015) model by a mixture of weights. Other models were also applied to learn multi-sense embeddings using external resources (e.g. WordNet) such as the work of (Chen et al., 2014). More recent models like SASI (Guo et al., 2019) uses an attention mechanism to select which sense is used in a word's context.

Distributed representations are based on the assumption that the meaning of a word depends on the context in which it occurs as stated by the Distributional Hypothesis (Harris, 1954). Indeed, the context is usually defined as the words which precede and follow the target word within some fixed window. Meanwhile, the meaning of a word is affected not only by its adjacent words but also by other words appearing with it and rules to combine them (i.e. compositionality). Moreover, the global meaning of sentences containing the target word can help determine its intended meaning. For example, let consider the word *present* in the following sentences: in "She sent me a present for my birthday" and in "There were 20 students present at the course". In the first sentence the word *present* means *gift* while the meaning of word present refers to *attendant* in the second sen-

Bouraoui, A., Jamoussi, S. and Ben Hamadou, A.
A Bi-recursive Auto-encoders for Learning Semantic Word Embedding.
DOI: 10.5220/0010716900003058
In *Proceedings of the 17th International Conference on Web Information Systems and Technologies (WEBIST 2021)*, pages 526-533
ISBN: 978-989-758-536-4; ISSN: 2184-3252

tence. Thus, sentencial context has impacts on words meaning. As each context word provide useful insights into the meaning of the target word, it would be desirable to capture the mutual interaction of distributed word vectors by a means of a compositional model. We assume that semantics in natural language is a compositional phenomenon where recursion is a natural manner of describing language.

Recently, deep neural models have been growing increasingly. Among them, auto-encoders have proven their robustness in representing the internal structure of several data. Thus, we opt to recursive models based on auto-encoders algorithm to learn semantic embeddings for words. Therefore, a question is raised: How we can learn contextual information to obtain an effective representation of the semantic of words based on recursive models. More precisely, how combine a word and its sentential context to represent its semantic by a recursive auto-encoders. So, we propose to separate the sentential context to left-side and right-side contexts. By such doing, the previous and future contextual information, around a target word, are incorporated to construct its semantic embedding. Certainly, when a person reads a text, he determines its meaning by recalling each inherent meaning a word can have and looking at the current context of its use. To emulate that, we initialize the embedding of the target word meaning when it appears by its old meaning to help detecting its new meanings. Therefore, this can provide an understanding of what the word means at a given point.

The remainder of this paper is organized as follows. Section 2 describes our proposed model for learning semantic word embedding. Section 3 presents the experimental setup followed by presenting and discussing results with comparison to state-of-the-art systems. Finally, this paper is concluded by providing some future work.

2 MODEL DESCRIPTION

In this section, we introduce our Bi-Recursive Auto-Encoders model for learning semantic embeddings of words. Our methodology consists firstly at constructing, for each word in the vocabulary, its set of contexts in the used corpus. In our work, we consider sentence as the usage context of a word. So, a set of contexts is the set of sentences containing the word of interest (target word). Then, we build the evolved semantic embedding of each target word over this set in an unsupervised manner.

Formally, we assume that a target word w_i can have a different sense for each sentence containing it. For example, if there are N sentences containing the word w_i, we will look to detect the evolved sense of the target word over these N sentences. Let us consider the target word $w_i \in V$ and we aim to learn its word sense embedding in some d-dimensional real space. V is the vocabulary of the considered words. C_i denotes the set of contexts (sentences) in which w_i occurs and $s_i \in \Re^d$ is the evolved semantic embedding of the target word. We apply our model to learn the evolved semantic embedding s_i of w_i over the set C_i. Our method randomly initializes embedding vectors of each word w_j and each target word sense vector $s_{i,t}$ (t refers to the occurrence number of a word in its set of sentences where $t \in 1,\ldots,|C_v|$ and $v \in 1\ldots|V|$), and updates these vectors during training. We use the constructed sense embedding $s_{i,t}$ over the context $C_{i,t}$ to initialize the sense embedding $s_{i,t+1}$ over the context $C_{i,t+1}$. Our methodology is illustrated in Figure 1.

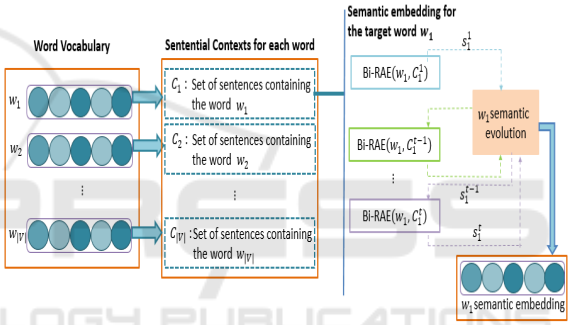

Figure 1: Proposed methodology.

Obviously, the meaning of a target word can be affected by the meaning of a word appearing before it as well as by the meaning of a word appearing after it. Additionally, we consider that the context words with closer distance to the target one influence considerably the word meaning. To simulate this observation, we split each sentence S into two sub-sequences of words around the target word. Thus, we construct two parts; left context and right context. Subsequently, we recursively encode the left context beginning by the first word in this sub-sequence of words to the closer word to the target. Similarly, we encode the right context beginning by the last word in this sub-sequence of words to the closer word to the target. Then, we combine these contextual semantics of the target word with itself to make its meaning more precise to obtain more meaningful word representation.

Figure 2 represents the architecture of our Bi-RAE model and its composition process to embed target word and their contextual words.

Given a sentential context, of a given target word

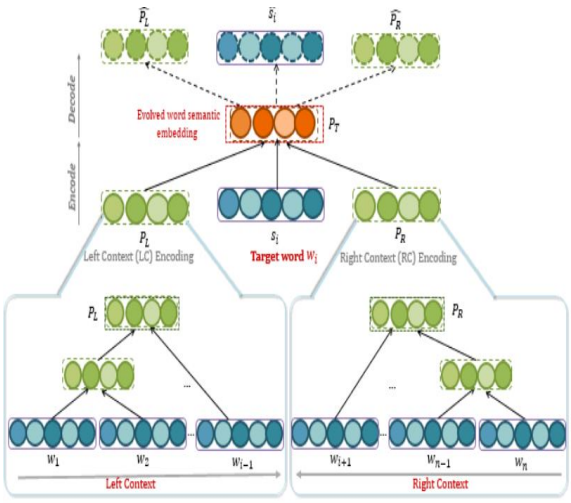

Figure 2: The proposed Bi-RAE deep model.

w_i, $S = [w_1, w_2, .., w_i, ..., w_n]$ with length n, our model regards it as a two separate sub-context: left context $[w_1, w_2, .., w_{i-1}]$ and right context $[w_{i+1}, ..., w_n]$. *LC* corresponds to the left context and *RC* designates the right context. To extract the contextual features from these sub-contexts, our model firstly encodes the left context *LC* using words from left to right and the right context *RC* from right to left. Intuitively, a context word can be informative for building the meaning of a target word in some contexts and less or not in others. Indeed, words that are closer to the target are, generally, more important. For example, in the sentence "I saw a cute grey [cat] playing in the garden", immediate neighbors (for the target word *cat*) are more informative than words at distance 3. Further, the relevance of the information from the word appearing before the target one decreases through recursion. In fact, at the target word position, the constructed embedding of the target word reflects its meaning influenced by its left context. When the recursive model continues reading through the sentence, the weight of the meaning of the left context words may decrease.
To illustrate more clearly our assumption, we carry out a concrete example. To represent the context of the target word "cat" in the sentence ("I saw a cute grey [cat] playing in the garden"), we encode the left context ("I saw a cute grey") and its right context ("playing in the garden"). If we continue the recursion process from the target word to the last word in the sentence from left to right, the model will gradually capture the semantic information of the whole sentence more than the semantic information of the target word. Further, the learned semantic information of the left context words may decrease. However, we aim to capture the relevant information in the sentential context, even when it is remote from the target word. Hence, to represent the context of a target word in a sentence, we separate the sentence into two sets of left-to-right and right-to-left context word embeddings around a target word.
For the target word representation learning, our model encodes the semantic of the left context words sequence and the right context words sequence.

Among each context words sequence, it combines each pair of sibling nodes into a potential parent node (hidden layer/latent representation). Considering the left context, our model takes the first pair of neighboring words vectors and defines them as potential children (c_1 and c_2) of a sub-sentential context $(c_1, c_2) = (w_1, w_2)$. Then, it concatenates and encodes them into a latent representation (parent vector) (p_1). Therefore, the network is shifted by one position and takes $(c_1, c_2) = (p_1, w_3)$ as input vectors. Afterwads, it computes a potential parent node (the next latent representation). This process is repeated until it reaches the last pair of vectors in the left context: $(c_1, c_2) = (p_{i-2}, w_{i-1})$. Each potential parent p_j is calculated using the following formula:

$$p_j = f(W_j^{\phi} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b_j^{\phi}), \tag{1}$$

where $W_j^{\phi} \in \Re^{d \times 2d}$ is an encoding weight matrix, c_1 and c_2 are the vectors corresponding to every child in the pair, b_j^{ϕ} corresponds to the encoding bias vector and f represents a non-linear activation function.

Therefore, we obtain left context encoding and right context encoding representations. Let P_L be the recursive encoding reading the words of a given sentence, appearing before a target word, from left to right; and P_R corresponds to the recursive encoding reading the words, appearing after the target word, from right to left. The concatenation of the left encoding representation P_L, the right encoding representation P_R and the target word representation s_i will be the input to an auto-encoder to build the latent semantic vector representation of the target word. We denote this parent node vector P_T.

$$P_T = f(W_T^{\phi} \begin{bmatrix} P_L \\ s_i \\ P_R \end{bmatrix} + b_T^{\phi}) \tag{2}$$

To obtain the embedding vector that encodes adequately the target word semantic, taking into account its left and right contexts, in the latent layer representation, we proceed to the decoding step. Our model reconstructs the entire spanned input underneath each node as shown in Figure 3.4. First, the model recon-

structs the input of P_T and computes:

$$\begin{bmatrix} \hat{P_L} \\ \hat{s_i} \\ \hat{P_R} \end{bmatrix} = f(W_T^{\varphi} P_T + b_T^{\varphi}) \quad (3)$$

Then, each parent node is decoded with the same structure of the encoding step. For example, it splits recursively the left context node $\hat{P_L}$ to produce vectors using this equation:

$$\begin{bmatrix} \hat{w_{i-1}} \\ \hat{p_{i-2}} \end{bmatrix} = f(W^{\varphi} \hat{P_L} + b^{\varphi}) \quad (4)$$

This formula is parametrized by W^{φ} which is a decoding weight matrix and b^{φ} which is a decoding bias vector.

At each parent node, the reconstruction error is calculated as the difference between the words vectors input in that node and its reconstructed counterparts. For a parent node p that spans words i to j, the reconstruction error is calculated as follows::

$$E_{rec}(p_{(i,j)}) = \left\| [w_i; ...; w_j] - [\hat{w_i}; ...; \hat{w_j}] \right\|^2 \quad (5)$$

3 EXPERIMENTS

In this subsection, we present our experiments and the achieved results using our deep model (Bi-RAE). We also detail some state-of-the-art methods and we compare their results with our deep model results.

3.1 Experimental Setup

We assess the quality of our Bi-RAE model on word similarity task as it is one of the most popular tasks for evaluating vector space models. We use the Stanford Contextual Word Similarity (SCWS) dataset (Huang et al., 2012), which is the most known dataset in which the task is to automatically estimate the semantic similarity of word pairs. It provides similarity scores between two words given their sentential context. This english dataset includes 2003 word pairs and their sentential contexts. It consists of 1328 noun-noun pairs, 399 verb-verb pairs, 140 verb-noun, 97 adjective-adjective, 30 noun-adjective, 9 verb-adjective, and 241 same-word pairs. For each pair of words, there are 10 human similarity scores provided (ranged in the interval $[0, 10]$). These scores are based on the word meanings in the context. Our Bi-RAE was implemented using the framework Tensorflow[1] and keras[2] for python. The input of our model is word embedding vectors initialized randomly.

We opted for $tanh(x) = f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ as activation function to encode the input representation and construct the latent representation. The same function was utilized to decode the latent representation and reconstruct the input representation. We chose *tanh* because the feature vector embedding of words includes positive as well as negative values within -1 and 1. We adopted the Adaptive Moment Estimation (Adam) optimizer (Kingma and Ba, 2015) to tune the network parameters to minimize the error reconstruction of our model. This choice is based on researchers' advice whose study in-depth different optimizers. For example, Sebastian Ruder developed a comprehensive review of modern descent optimization algorithms (Sebastian, 2016) and recommended Adam as the best overall choice. Besides, Andrej recommended in their Stanford course on deep learning[3]. Furthermore, we empirically tested some other optimizers like SGD and RMSprop and we remarked that Adam provided the best results. We set the initial learning rate to 0.001 as recommended in (Kingma and Ba, 2015).

The evaluation procedure consists in measuring how appropriately the automatically-obtained similarity scores on word pairs of the benchmark match the ratings produced by humans. In order to compute the similarity of two words using our sense embeddings, we used the cosine similarity measure. Then, we measured how much does the model estimation of word similarity resembles that of the human judgment by computing the Spearman's correlation metric. Finally, we report the GlobalSim and AvgSim similarity metrics (Reisinger and Mooney, 2010) as evaluation metrics.

3.2 Hyper-parameter Tuning

To train a model that can achieve the best performance, suitable hyper-parameters tuning is required. Among these hyper-parameters, we can mention the dimension of word embeddings (embedding size) and the number of epochs. We released many experiments to choose the best values of these hyper-parameters and find ultimately the parameters values that allow good convergence of the model to the best results. To do so, we started by adopting a set of values for each parameter. Therefore, we varied the parameters

[1] https://www.tensorflow.org/learn

[2] https://keras.io

[3] http://cs231n.github.io/

and computed the Spearman score for each parameter value.

To study the impact of the vector dimension on the performance of the proposed model, we built word embeddings using different dimensionalities ranging in [50,100,150,200,300,400,500] while fixing the number of epochs at 50. Afterward, we selected the dimension size that maximized the Spearman correlation score.

In Figure 3, we report Spearman values obtained with regard to different embedding sizes.

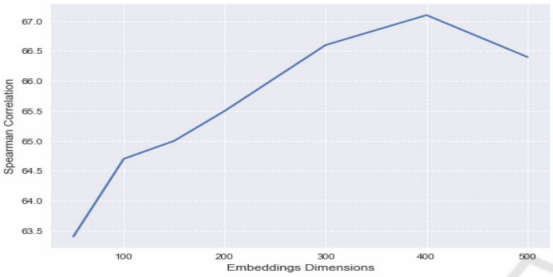

Figure 3: Obtained Spearman values multiplied by 100 when varying the embedding dimension.

Figure 3 shows the impact of the embedding dimension on the results obtained by the proposed model. Overall, the Bi-RAE model is able to learn accurate word embeddings even with a dimension of 50. In addition, the performance increases gradually with the embedding size, reaching a peak when the dimension is equal to 400. Thus, we set the dimension size at 400.

After choosing the best dimension for word embeddings, we go forward to study the effect of the numbers of epochs on the Spearman correlation. To do so, we varied the number of epochs in the ranges of [10,50,100,200,300,400,500].

Figure 3 shows the Spearman correlation performance of our model for different numbers of epochs while keeping the dimension of embedding constant. It is clear, from this figure, that the performance increases by rising the number of epochs which ranges from 1 to 100 then it decreases continuously and nearly keeps steady after 200 iterations. According to this experiment, the number of epochs was set at 100.

3.3 Results and Discussions

In order to evaluate and position our Bi-RAE, we compare it against several baseline methods. These methods include single word embeddings techniques and multi-prototype word embeddings. They were chosen as baselines because they present well-known state-of-the-art methods for word embeddings.

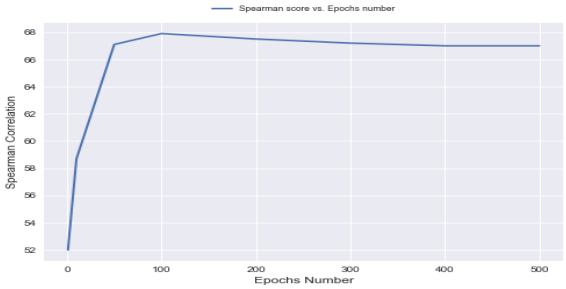

Figure 4: Spearman correlation of our Bi-RAE model vs epochs number.

- **Word2Vec Model:** We performed our own training on the used dataset utilizing the implementation of Word2Vec from the library Gensim[4]. This algorithm uses a one hidden layer neural network to predict a word by considering its context (CBOW version of the algorithm).
- **Glove Model:** It is an alternative way to learn word embeddings. Training is was carried out on aggregated global word-word co-occurrences statistics from corpus. We trained this model on the used dataset using the implementation of Glove from the library Gensim[5].
- **C&W Model:** Authors of (Collobert et al., 2011) proposed an embedding model using CNN architecture designed to preserve more information about word features relations. Indeed, they extracted local feature vectors using a convolutional layer. Then, they combined these features employing a pooling operation in order to obtain a global features vector. The pooling operation is a max-pooling operation which forces the network to capture the most useful local features produced by the convolutional layer.
- **EH Model:** This method was introduced in (Huang et al., 2012). It consists in generating, at first, single-sense word embeddings and computing out the context embeddings. Then, a clustering step of these context embeddings was performed. The obtained results were used to re-label each occurrence of each word in the corpus. In addition, authors applied their model to the labeled corpus to generate the multi-vector word sense embeddings.
- **MSSG, MSSG-NP Models:** Authors of (Neelakantan et al., 2015) improved multi-sense word embeddings model by dropping the assumption that each word should have the same number of

[4]https://radimrehurek.com/gensim/models/Word2Vec.html

[5]https://textminingonline.com/getting-started-with-Word2Vec-and-glove-in-python

senses. They proposed a non-parametric model to automatically discover a varying number of senses per word. More precisely, NP-MSSG measures the distance of the current word to each sense, picks up the nearest one and learning its embedding via a standard skip-gram model.

- **Li & Ju. Model:** Authors of (Jiwei and Dan, 2015) used a similar strategy to (Neelakantan et al., 2015) by integrating the Chinese Restaurant Process[6] into the Skip-gram model. Indeed, they employed this process to determine the sense of a word and learn the sense embedding.
- **Chen Model:** (Chen et al., 2014) utilized glosses in WordNet as clues to learn the distributed representation of word sense. Then, they represented each word sense by the vector averaged over all the words occurring in the corresponding gloss.
- **SASI Model:** Authors of (Guo et al., 2019) proposed the SASI model as an extension to Word2Vec to learn multi-vector embeddings for word meaning. The SASI model uses an attention mechanism to select which sense is used in a token's context, which they contrast to alternatives for sense selection.
- **TF-IDF:** It is composed of two parts: TF which is the term frequency of a word, i.e. the count of the word occurring in a document and IDF, which is the inverse document frequency, i.e. the weight component that gives higher weight to words occurring in only a few documents.
- **Pruned TF-IDF (Huang et al., 2012):** Frequency-based pruning uses term frequency information to measure the importance of the terms and to prune relatively unimportant terms or segments of the document. Pruned TF-IDF consists in pruning the low-value TF-IDF features.

We executed the two most popular single vector word embedding methods (Word2Vec and Glove) on the SCWS dataset using 50, 300 and 400 as embedding dimensions.

In Table 1, we report the Spearman correlation scores p between the embedding similarities and human judgments using our proposed model, Word2Vec and Glove models. Furthermore, we illustrate the results obtained with two others single prototype word embedding C&W (Collobert et al., 2011) and TF-IDF (Huang et al., 2012).

[6]https://www.statisticshowto.com/chinese-restaurant-process/

Table 1: Bi-RAE performance, multiplied by 100, on SCWS in comparison with those of Word2Vec, Glove,C&W and TF-IDF.

Method	**Spearman** $p \times 100$
Word2Vec (50 dim)	48.6
Word2Vec (300 dim)	50.3
Word2Vec (400 dim)	50.7
Glove (50 dim)	44.8
Glove (300 dim)	46.3
Glove (400 dim)	46.9
C&w (Collobert et al., 2011)	57.0
TF-IDF (Huang et al., 2012)	26.3
Bi-RAE (50 dim)	64.9
Bi-RAE (300 dim)	67.2
Bi-RAE (400 dim)	67.9

Experimental results show the potential of the Bi-RAE model to reach good Spearman scores outperformed the two baseline models (Word2Vec and Glove) for different dimensions. In fact, it reach Spearman scores equal to 64.9, using 50 as embedding dimension, 67.2 using 300 as embedding dimension and 67.9 using 400 as embedding dimension. These results indicate that the quality of our word embeddings is better than those obtained by Word2Vec and Glove. They show the clear benefit of learning word semantic embedding using its left/right contexts. Thus, the gap in Spearman score between the Bi-RAE model and baselines is great. Indeed, the full model has to be fed with each whole sentence to get the word representations, unlike Word2Vec vector representations, which are constant regardless of their context.

Table 2 shows the performance of our proposed model while comparing it against most popular multi-prototype models evaluated on the SCWS dataset. As word representation with multiple prototypes is used to build multiple distinct vectors for all senses of a word, (Reisinger and Mooney, 2010) presented the AvgSim measure and the GlobalSim, to compute similarities. GlobalSim uses one representation per word to compute similarities, while AvgSim calculates the similarity employing different embeddings per word based on the context information. Spearman correlation scores p computed between the embedding similarities and human judgments is used. In Table 3, we report these two measures for multi-prototype word embeddings. We also present the results obtained applying the word distributional representations Pruned TF-IDF (Huang et al., 2012).

It is observed from Table 3.4 that our model Bi-RAE achieved a better performance compared to baseline techniques (Collobert et al., 2011; Huang et al., 2012; Chen et al., 2014; Neelakantan et al.,

Table 2: Spearman correlation scores, multiplied by 100, on the SCWS dataset in comparison with prior work. Fields marked with "'-"' indicate that the results are not available.

Method	**Spearman** $p \times 100$	
	AvgSim	GlobalSim
Pruned TF-IDF (Huang et al., 2012)	60.4	62.5
EH Model (50 dim) (Huang et al., 2012)	62.8	58.6
MSSG (300 dim) (Neelakantan et al., 2015)	67.2	65.3
MSSG-NP (300 dim) (Neelakantan et al., 2015)	67.3	65.5
MSSG (50 dim) (Neelakantan et al., 2015)	64.2	62.1
MSSG-NP (50 dim) (Neelakantan et al., 2015)	64.0	62.3
Li&Ju. (300 dim) (Jiwei and Dan, 2015)	66.4	64.6
Chen (200 dim)(Chen et al., 2014)	66.2	64.2
SASI (300 dim) (Guo et al., 2019)	64.8	-
Bi-RAE (50 dim)	64.9	
Bi-RAE (300 dim)	67.2	
Bi-RAE (400 dim)	67.9	

2015; Jiwei and Dan, 2015; Guo et al., 2019) for either 50 or 300 vector dimensions. Our model attained a Spearman correlation score equal to 67.2 using 300 as dimension size and 67.9 using 400 dimension size. This finding indicates that the recursive composition (based on auto-encoders) strategy proposed in our work is beneficial. Furthermore, it shows again that our strategy of modeling word sense embeddings by means of their right/left sub-sentential context embeddings is helpful to improve the quality of the word sense vector. Additionally, the Bi-RAE model treats the left sub-sentential context, from first word to the target word, and the right sub-sentential context, from last word to target word, to mine the semantics of the target word. This strategy allows extracting the relations between words that are far from the target while affect its semantic. Besides, obtaining Spearman scores competitive to those obtaining by AvgSim measure for multi-prototype word embeddings indicates that computing the word sense embeddings as evolved vector taking into account its sentential contexts is promising. It worth mention that our results are obtained in unsupervised framework without using any extra-information.

Recall that representing word meaning by multi-vectors requires either sense inventories or clustering methods. The latter carried out a form of word sense discrimination as a pre-processing step by clustering contexts for each word, which ignores complicated correlations among words as well as their contexts (Pengfei et al., 2015). Added to that, a common strand of most unsupervised models is that they extend the SkipGram model enable the capture of sense-specific distinctions. These models represent the context of a word as the centroid of it words' vectors and clusters them to form the target word's sense representation. During training, the intended sense for each word is dynamically selected as the closest sense to the context and weights are updated only for that sense. Hence, sense embedding is conditioned on the word embeddings of its context. Therefore, the context in these models is represented by a vector composed of the occurrences of the neighboring words or a weighted average of the surrounding words of the target word. Such context definitions neglect the relative order of words in the context window and the rules of combining words, which influences the quality of the representations based on such context representations. Thus, these models pay equivalent attention to the words to the left and to the right of the target word. In addition, these models set an equal number of senses for each word which is not real. On the contrary, the Bi-RAE model provides representations of words semantic which depend on the sentential context. Moreover, the relative importance of left or right contexts may in principle depend on the linguistic properties of the corpus language, in particular its word ordering constraints. In this vein, our model allows us to take account such effects that context has on word semantic. It allows extracting useful information about the semantic of a target word based on its ordered sub-sequences of left-side and right-side contexts using recursive auto-encoders. This combination strategy of word context of the Bi-RAE model leads to improved results. This means that the Bi-RAE model is able to encode correlations among words as well as their contexts. Further, in contrast to other approaches, the proposed deep model Bi-RAE constructs an evolved vector to embed words meaning by taking into account the sentential context, separated to left/right context, of the target word. This evolved vector adapt the semantics

of a target word based on its context. This idea was inspired by the fact that most sentences use a single sense per word and human can determine the meaning of a word in a given context by referring to the different meanings already-known of this word. For this reason, we construct a new sense of a word using its previously-learned sense by considering its previous context. Our overall process allows us to achieve encouraging results.

4 CONCLUSIONS

In this paper, we present a new deep model, named Bi-RAE (Bi-Recursive Auto-Encoders), to learn word meaning embedding from scratch. This model aims to construct a dense informative representation of word meaning using its sentences as context. More precisely, it treats each sentence containing the target word as two sub-contexts (left and right contexts around the target). Our model is based on the idea of learning dynamically an evolved semantic embedding of a word relying on the words contained in their sentential context and its latest semantic representation. Thus, it was possible to create semantic embeddings of words that captures as accurate as possible the meaning of the word conveyed in their contexts. We released experiments on a very challenging task in NLP; the semantic similarity task. Experimental results proved the effectiveness of our unsupervised model compared to well-known methods modeling word semantic embeddings using either in single or multi prototypes. In our future work, we we would couple our proposed model with an attention mechanism to further improve its learned embeddings.

REFERENCES

Chen, X., Liu, Z., , and Sun, M. (2014). A unified model for word sense representation and disambiguation. In *Proceeding of the Conference on Empirical Methods in Natural Language Processing*, pages 1025–1035.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Fei, T., Hanjun, D., Jiang, B., Bin, G., Rui, Z., Enhong, C., and Tie-Yan, L. (2014). A probabilistic model for learning multi-prototype word embeddings. In *Proceeding of the International Conference on Computational Linguistics*, pages 151–160.

Guo, F., Iyyer, M., and Boyd-Graber, J. L. (2019). Inducing and embedding senses with scaled gumbel softmax. *ArXiv*.

Harris, Z. (1954). Distributional structure. *Word*, 10(23):146–162.

Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 1: Long Papers*, pages 873–882.

Jiwei, L. and Dan, J. (2015). Do multi-sense embeddings improve natural language understanding? In *Empirical Methods in Natural Language Processing*, pages 1722–1732.

Kingma, D. P. and Ba, J. (2015). Adam: a method for stochastic optimization. In *Proceeding of the International Conference on Learning Representations*, pages 1–13.

Liu, Y., Liu, Z., Chua, T., and Sun, M. (2015). Topical word embeddings. In *Proceedings of the Twenty-Ninth Association for the Advancement of Artificial Intelligence Conference, January 25-30, 2015, Austin, Texas, USA.*, pages 2418–2424.

Mikolov, T., tau Yih, S. W., and Zweig, G. (2013). Linguistic regularities in continous space word representations. pages 746–751. Proceeding of the Annual Conference of the North American Chapter of the Association for Computational Linguistics.

Neelakantan, A., Shankar, J., Passos, A., and McCallum, A. (2015). Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the Association for Computational Linguistics*, pages 1059–1069.

Nguyen, D. Q., Nguyen, D. Q., Modi, A., Thater, S., and Pinkal, M. (2017). A mixture model for learning multi-sense word embeddings. In **SEM*, pages 121–127. Association for Computational Linguistics.

Pengfei, L., Xipeng, Q., and Xuanjing, H. (2015). Learning context-sensitive word embeddings with neural tensor skip-gram model. In *Proceeding of the International Joint Conference on Artificial Intelligence*.

Reisinger, J. and Mooney, R. J. (2010). Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics.

Sebastian, R. (2016). An overview of gradient descent optimization algorithms. *Computing Research Repository*, abs/1609.04747.