Improving Word Embeddings for Antonym Detection Using Thesauri and SentiWordNet

Zehao Dou(✉), Wei Wei, and Xiaojun Wan

Peking University, Beijing, China
{zehaodou,weiwei,wanxiaojun}@pku.edu.cn

Abstract. Word embedding is a distributed representation of words in a vector space. It involves a mathematical embedding from a space with one dimension per word to a continuous vector space with much lower dimension. It performs well on tasks including synonym and hyponym detection by grouping similar words. However, most existing word embeddings are insensitive to antonyms, since they are trained based on word distributions in a large amount of text data, where antonyms usually have similar contexts. To generate word embeddings that are capable of detecting antonyms, we firstly modify the objective function of Skip-Gram model, and then utilize the supervised synonym and antonym information in thesauri as well as the sentiment information of each word in SentiWordNet. We conduct evaluations on three relevant tasks, namely GRE antonym detection, word similarity, and semantic textual similarity. The experiment results show that our antonym-sensitive embedding outperforms common word embeddings in these tasks, demonstrating the efficacy of our methods.

Keywords: Antonym detection · Word embedding · Thesauri
SentiWordNet

1 Introduction

Word embedding plays an important role in word representation since it effectively captures semantic information of words. A good embedding provides vector representations of words such that the relationship between two vectors mirrors the linguistic relationship between the two words. Such distributed representations of words in a vector space contribute to achieving better performance in many natural language processing tasks such as text classification [7], abstraction generation, and entity recognition.

By grouping similar words, the existing word embeddings perform well on synonyms, hyponyms, and analogies detection. However, most of them are insensitive to antonyms, since they are trained based on the distributional hypothesis [4] and word distributions in a large amount of text data, where antonyms usually have similar contexts. To be specific, a pair of antonyms, for example, "long"

M. Zhang et al. (Eds.): NLPCC 2018, LNAI 11109, pp. 67–79, 2018.
https://doi.org/10.1007/978-3-319-99501-4_6

and "short", tend to appear in similar context environment. This leads to the serious problem that it is extremely difficult to discriminate antonyms from synonyms. It is important to solve this problem and obtain antonym-sensitive word embedding, since such embedding has the potential to make contribution in some certain tasks such as semantic textual similarity.

A key characteristic of current word embeddings attracts our attention: the vectors of related words are supposed to have close directions while unrelated words correspond to vectors in opposite directions. Therefore, the cosine distance between completely unrelated words is close to -1, while the cosine distance between a pair of related words, including both synonyms and antonyms, is close to 1. In other words, instead of evaluating sematic similarity, these embeddings evaluate the relatedness between words, treating synonyms and antonyms equally without discrimination.

To obtain antonym-sensitive word embedding, we start from modifying the above characteristic. In our model, the cosine distance between related words is supposed to be either close to 1 or close to -1, while the cosine distance between unrelated words is close to 0. Then in the vector space, the related words of a particular word distribute in two areas, either around the position of the word or around the head of the reversed vector. In this situation, we can design a more reasonable model to detect antonyms. Given that both antonyms and synonyms are highly-related word pairs, our goal is to make the cosine distance of synonyms close to 1 and the cosine distance of antonyms close to -1. Then a pair of words with a negative cosine distance close to -1 are likely to be antonyms.

With this idea, we propose a novel approach to train our antonym-sensitive embedding, named Word Embedding Using Thesauri and SentiWordNet with Distributional Corpus-based Information (WE-TSD). Firstly, we modify the objective function of Skip-Gram model in order to achieve the goal stated in the previous paragraph. Secondly, our model uses thesauri information to get supervised synonym and antonym word pairs, and we also utilize SentiWordNet provided by [1] to make sentimental analysis of every word in vocabulary. SentiWordNet is a dataset which labels each English word with a corresponding 3-dimensional vector. The three components of the vector respectively represent the positive emotion rate, negative emotion rate and objective rate of the word with their sum 1.

To demonstrate the efficacy and task adaptability of our antonym-sensitive embedding, we conduct evaluations on three relevant tasks, namely GRE antonym detection, word similarity, and semantic textual similarity. The experiment results show that our antonym-sensitive embedding outperforms common word embeddings in all these tasks, convincingly demonstrating the effectiveness of our methods.

2 Related Works

In the past decades, research on natural language processing has made great success, where a good word representation plays an crucial role. At first, one-hot

word vector has been widely used in the bag-of-words (BOW) text model. The success of text categorization [5] with BOW popularized this model. However, one-hot vector has a serious weak point: with each word represented as a completely independent entity, this word representation hardly provides any notion of similarity.

To reduce the size of vector space and encode the relationship between words, Rumelhart et al. [17] described a new learning procedure to learn representations. With the word embedding model, many natural language processing tasks, such as entity recognition and dependency parsing, gained second life. In this word embedding model, semantic relationship between words are well encoded and can be easily detected.

However, the huge time cost and computational complexity became an obstacle, leading researchers to find a more efficient and less complex model. In 2013, two highly efficient models were proposed by Mikolov et al. [9], namely CBOW (continuous bag-of-words) and Skip-Gram. CBOW aims to predict a center word from the surrounding context in terms of word vectors. Skip-gram does the opposite, and predicts the distribution of context words from a center word. With these two models, we are able to learn word embeddings with large corpus.

On antonym detection tasks, Polarity Inducting Latent semantic Analysis proposed by Yih et al. abstracts polarity information from thesauri and they use context vectors to cover those words out of thesauri vocabulary. Ono et al. [14] proposes a word embedding-based antonym detection using thesauri and distributional information, which combined the traditional Skip Gram objective function [8] and the polarity thesauri information. This model contributes to finding a balance between the Skip Gram model and WE-T (Word Embedding with Thesauri Information) model. Nguyen et al. [13] proposes a novel model which integrates distributional lexical contrast into word embedding. However, the results of these previous work are far from perfection, motivating us to conduct the research in this paper and make some improvement.

3 Our Approach to Obtain Antonym-Sensitive Word Embeddings

3.1 Skip-Gram Model and Our Modification

In this section, we firstly introduce the original Skip-Gram Model and negative sampling proposed by Mikolov et al. [8], which is one of the most popular methods to train word embeddings. Next, to adapt for our antonym-sensitive embedding, we make some modification on the objective function in the Skip-Gram Model.

Skip-Gram Model with Negative Sampling. A word embedding is a mapping $V \rightarrow R^D$ that maps a word to its corresponding D-dimensional word vector. This is called a D-dimensional word embedding. The most widely used

approaches of training word embeddings are Skip-Gram and CBOW [8]. In this paper, we mainly focus on the former one.

In the standard Skip-Gram Model, we aim to obtain a word embedding that can predict the context words around a given target center word effectively. To be specific, our goal is to maximize the following objective function,

$$\frac{1}{T}\sum_{t=1}^{T}(\sum_{t-c\leq i\leq t+c, i\neq t}\log p(w_i|w_t)), \tag{1}$$

where $w_1, w_2, \cdots w_T$ are the words in the whole training corpus, and c is a hyperparameter corresponding to window size, determining the number of context words induced by the target center word w_t. The most essential part of this model is the conditional probability $p(w_i|w_t)$. Following Mikolov et al.'s idea [9], the conditional probability is expressed as follows,

$$p(w_i|w_t) = \frac{e^{u'^{T}_{w_i} u_{w_t}}}{\sum_{j=1}^{|W|} e^{u'^{T}_{w_j} u_{w_t}}}, \tag{2}$$

where u'_w, u_w denote the representations of word w respectively as context and target word. W is the vocabulary set extracted from the training corpus and $|W|$ denotes the vocabulary size. Further, Eq. (2) can also be written as

$$p(w_i|w_t) = softmax\left(u'^{T}_{w_i} u_{w_t}\right). \tag{3}$$

Although the basic model described above seems reasonable, its performance is actually not satisfying enough. Due to the normalization term, the time complexity of the above conditional probability equation is $O(|W|)$, which is unacceptable especially when W is large. Therefore, two modifications have been proposed. Firstly, Mikolov et al. [8] present hierarchical softmax as a much more efficient alternative to the normal softmax. With hierarchical softmax, the time complexity can be reduced to $O(\log(|W|))$.

Another idea to improve the efficacy of word embeddings and reduce the training cost is negative sampling, which is also provided by Mikolov et al. [8]. For every training step, instead of looping over the entire vocabulary, we just sample several negative examples. Although negative sampling is based on the Skip-Gram model, it is in fact optimizing a different objective. The new objective function tries to maximize the probability of a word and context being in the corpus data if it indeed is, and maximize the probability of a word and context no being in the corpus data if it indeed is not, which is shown as follows,

$$L = \sum_{w\in W}(\sum_{t-c\leq i\leq t+c, i\neq t}\log(\sigma(v'^{T}_{w} v_{w_i})) + \sum_{u\in NEG(w)}\log(\sigma(-v'^{T}_{w} v_u))), \tag{4}$$

where σ denotes the sigmoid function, and $NEG(w)$ denotes a small subset of all the negative examples of target word w sampled from a modified unigram distribution [8]. The number of components in the negative sampling subset is

called "negative size". Besides, we employ the subsampling [8] which discards the words according to the following probability:

$$P(w) = 1 - \sqrt{\frac{t}{p(w)}}, \tag{5}$$

where t is a threshold to control the discard action and $p(w)$ is the occurrence probability of w in the training corpus. Subsampling is very useful and it aims to make less frequent words be sampled more often.

Modified Skip-Gram Model. However, the widely used Skip-Gram Model described above can not fulfill our goal. In the result embeddings, the cosine distance between unrelated words is close to -1, while the cosine distance between a pair of related words, including both synonyms and antonyms, is close to 1. In other words, instead of evaluating sematic similarity, these embeddings evaluate the relatedness between words, treating synonyms and antonyms equally without discrimination.

To obtain antonym-sensitive word embedding, we want the cosine distance between related words to be either close to 1 or close to -1, while the cosine distance between unrelated words to be close to 0. Then in the vector space, the related words of a particular word distribute in two areas, either around the position of the word or around the head of the reversed vector. Next, given that both antonyms and synonyms are highly-related word pairs, our goal is to make the cosine distance of synonyms close to 1 and the cosine distance of antonyms close to -1.

Therefore, in our model, the objective function with distributional information of unsupervised training corpus should be:

$$Func1 = \sum_{w \in C} \Big(\sum_{t-c \le i \le t+c, i \ne t} \log(\sigma((v'^T_w v_{w_i})^2)) + \sum_{u \in NEG(w)} \log(\sigma(-(v'^T_w v_u)^2)) \Big). \tag{6}$$

Our modification is the square functions inside the sigmoid function σ. With the square functions, the absolute value of the dot product of positive samples and target word will get closer to 1 during training, which means their cosine distance will be either close to 1 or close to -1. This result fulfills our expectation.

The modified Skip-Gram Model with the new objective function (6) plays an important role in our final model. However, both antonym and synonym are considered "related" and we still can not discriminate antonyms from synonyms based on their cosine distance. Therefore, we need to utilize supervised dataset including Thesauri Dataset and SentiWordNet, in order to make the cosine distance between synonyms close to 1 while the cosine distance between antonyms close to -1. As a result, the lower their cosine distance is, the more likely to be antonyms they are.

3.2 Word Embedding Injecting Thesauri Dataset Information with Max Margin Framework (WE-TM)

In this section, we introduce a sub-model using thesauri dataset information. Following Ono et al.'s work [14], we are going to embed all the words in thesauri into vectors.

According to our target, we need to increase the dot product between synonyms and decrease the dot product between antonyms. Therefore, we set up an objective function as shown below,

$$Func2 = -\sum_{w \in V} max(0, \gamma - \frac{1}{|S(w)|} \sum_{s \in SYN(w)} sim(w, s) + \frac{1}{|A(w)|} \sum_{a \in AYN(w)} sim(w, a)), \tag{7}$$

where V denotes the vocabulary in thesauri; $SYN(w)$ denotes all the synonyms of word w and $ANT(w)$ denotes all the antonyms of w. $|S(w)|$ and $|A(w)|$ denote the sizes of $SYN(w)$ and $ANT(w)$. The hyper-parameter γ will be set later. $sim(w, s)$ denotes the similarity of the two words in our model, which can be mathematically expressed as:

$$sim(w, s) = v'^{T}_{w} v_s. \tag{8}$$

The maximization of $Func2$ makes the similarity score between synonyms very high and that between antonyms very low. Besides, for some indirect antonyms, for example, "beautiful" and "bitter", although they are not antonyms according to thesauri information, their similarity score will also be relatively low because "beautiful" is the synonym of "nice" and "bitter" is the antonym of "nice". The directions of word vectors of "beautiful" and "nice" are almost the same while the directions of word vectors of "nice" and "bitter" are almost opposite. This sub-model is reasonable and effective, and we name this model WE-TM (Word Embedding using Thesauri Dataset Information with Max Margin Framework). Comparing with the WE-TD model proposed by [14], our model introduces the Max Margin Framework which can significantly decrease the risk of overfitting.

3.3 Word Embeddings Based on SentiWordNet (WE-S)

Besides thesauri, the other supervised knowledge base we are going to use is SentiWordNet [1], which is explicitly devised for supporting sentiment classification and opinion mining applications. It is a lexical resource in which each WORDNET synset w is associated to three numerical scores $[Pos(w), Neg(w), Obj(w)]$, describing how objective, positive, and negative the terms contained in the synset are. The triple has the following property:

$$Pos(w) + Neg(w) + Obj(w) = 1. \tag{9}$$

From this dataset, we need to abstract some senti-synonym word pairs and some senti-antonym word pairs according to their corresponding sentiment triple.

We hope the senti-synonym word pairs have high word similarity scores and senti-antonym word pairs have low word similarity scores, just like those in thesauri information. Firstly, we drop all the synsets with the sentiment triple $[0, 0, 1]$ because words in these categories are completely objective and we can hardly conduct any sentimental analysis on them. Then we define a concept named senti-similarity, which evaluates the degree one word is similar to another in respect of sentimental inclination. Mathematically, it is expressed as follows,

$$SentiSim(w_1, w_2) = \frac{Pos(w_1)Pos(w_2) + Neg(w_1)Neg(w_2)}{\sqrt{((Pos(w_1)^2 + Neg(w_1)^2)(Pos(w_2)^2 + Neg(w_2)^2)}}. \tag{10}$$

In this expression, we ignore the objective judgement ratio $Obj(w)$ of these words and calculate the normalized dot product of $[Pos(w_1), Neg(w_1)]$ and $[Pos(w_2), Neg(w_2)]$. The higher the senti-similarity is, the more similarly these two words express in sentimental inclination. In our model, we think that those senti-synonyms should have not only high senti-similarity score but also high word similarity score and vice versa. So our objective function is:

$$Func3 = \sum_{w_1, w_2 \in SWN} (SentiSim(w_1, w_2) - \overline{SentiSim}) \cdot \log \sigma(sim(w_1, w_2)), \tag{11}$$

where SWN denotes the vocabulary of SentiWordNet, and $\overline{SentiSim}$ denotes the average senti-similarity value of all word pairs in SWN; σ is the sigmoid function. Through the maximization of $Func3$, word pairs with higher senti-similarity score will also have higher word similarity.

3.4 Our Approach(WE-TSD)

Our final model is the integration of the three sub-models above. Our objective function is:

$$Func = Func1 + c_1 Func2 + c_2 Func3, \tag{12}$$

and our goal is to maximize the function. All the three sub-models are necessary. The first model Modified Skip-Gram makes use of distributional corpus information and describes the relatedness of words based on unsupervised corpus. The WE-T model controls the word similarity between synonyms and antonyms, and the WE-S model controls the similarity between senti-antonyms and senti-synonyms. The vocabulary size of WE-S is a lot larger than WE-T and it almost covers the whole vocabulary. Each of the three models has its own effect and none of them can be discarded.

In our objective function, c_1 and c_2 are two coefficients used to balance the importance of the three sub-models. While conducting experiments, we need to tune the coefficients and parameters to achieve better performance. We call this novel model WE-TSD which means Word Embeddings Based on Thesauri, SentiWordNet and Distributional information.

4 Experiments

4.1 Evaluation Settings

In this section, we introduce three relevant tasks which we utilize to evaluate our methods, namely GRE antonym detection task, word similarity task and textual similarity task. Both the dataset and the evaluation metrics are described in detail below. In our experiments, we not only compare our model with several baselines but also with some advanced embeddings including WE-T (Word Embedding using Thesauri only), WE-TD (Word Embedding using Thesauri and discributional condition [14]) and WE-S (Word Embedding using SentiWordNet only).

GRE Antonym Detection Task. GRE antonym questions dataset is widely used in antonym detection tasks which is originally provided by [11]. It is a set of questionnaires with several hundreds of single-choice questions. Each question has a target word and five candidates. We need to choose the only antonym of the target word in the five candidates. Moreover, this dataset is divided into two parts, development set which contains 162 questions and test set which contains 950 questions. Since there are 160 questions appear in both of the two sets, we will report results on both the whole test set and the remaining test set with 790 (= 950 −160) questions just like [14] do in their evaluation experiment of WE-TD model.

When we evaluate our model on these single-choice questions, we calculate the similarity of the target word and the five candidates one by one, and then the candidate who has the lowest similarity with the target word is declared the winner. After finishing all the questions, we evaluate our embedding by F-score following Zhang et al. (2014). If our model doesn't contain the target word or the five candidates, the question will be left unanswered. Unanswered questions are regarded the same as wrong-answered.

Word and Semantic Text Similarity Task. In word similarity experiments, one of the most widely used dataset is WordSim353 provided by [3]. In this experiment, we use dataset WordSim353. Since this dataset consists of two parts, the relatedness part and the similarity part, we conduct our experiment on the two parts respectively. Obtaining three results on WordSim353 (Rel), WordSim353 (Sim) and WordSim353 (Combined), we can show the effectiveness of our model comprehensively. In WordSim353, there are 353 word pairs and a human labeled word similarity score for each word pair.

We compute the similarity of these word pairs according to the absolute value of the cosine distance of their corresponding word vectors. The higher the calculated value is, the more related or similar the word pair is supposed to be. Then, we calculate the Spearman correlation between the word similarity scores from our model and the human labeled scores for comparison.

Semantic Textual Similarity (STS) is the task of determining the degree of semantic similarity between two sentences. STS task is an important foundation of many natural language processing applications, but the performance still remains to be improved. One of the difficulties is that common systems are insensitive to antonyms. Two sentences with high overlap but also a pair of antonyms usually indicate opposite meanings, but common systems tend to generate a high similarity score. Therefore, our antonym-sensitive embedding has a great potential to improve the results by avoiding such errors.

We conduct experiments on STS Benchmark [2], which comprises a selection of the English datasets used in the STS tasks organized in the context of SemEval between 2012 and 2017. The performance is measured by the Pearson correlation of machine scores with human judgments. We build a convolutional neural network, which achieves best results on this dataset, as described in [18], and use different word embeddings as the input of the model.

4.2 Training Resource and Parameter Settings of Our Model

Our supervised datasets used to train our WE-TSD model include two parts. The first one is antonym and synonym pairs in two thesauri, WordNet provided by [10] and Roget provided by [6], the other is SentiWordNet provided by [1]. The unsupervised training corpus comes from Wikipedia. We lowercase all the words and drop all the stopping words and punctuations. The size of raw text is over 10GB and the huge size of unsupervised dataset helps us to train the word embedding more accurately.

When training our WE-TSD model, the dimension of embeddings is set to 300, the negative size is 10, window size is 5, and the threshold for subsampling was 10^{-8}. We utilize Adam as the optimizer. During training, we use learning rate decay to fit the training corpus, and the number of iteration epochs is set to 50. In Func2, $\gamma = 0.6$. The parameter c_1 is 100, c_2 was 2.5. While determining these two hyper-parameters, we take the proportion of the size of Wiki Corpus, Thesauri and SentiWordNet vocabulary into account.

4.3 Results of Experiments

GRE Antonym Detection Task. In experiments, we compare our model with baselines including Encarta lookup from [?], S2Net from [?], WordNet & Roget BFTP from [12], WE-T and WE-TD from [14] and so on. Since we evaluate on the same data as [14], we simply report the evaluation results reported by them.

In the GRE Antonym Detection Task, two models obviously surpass the others, namely **WE-TD** proposed by [14] and our **WE-TSD**. Both these two models make use of supervised information about synonyms and antonyms, which may play an important role in this task. The major difference between **WE-TD** and our **WE-TSD** is our utilization of SentiWordNet. On the complete test set **TestSet(950)**, we get an F-score of 92% and outperform the **WE-TD** model which achieves an F-score of 89%, demonstrating the contribution of the semantic information in SentiWordNet.

More detailed results are listed in Table 1. Obtaining the state-of-the-art results, we can state that our antonym-sensitive embedding is capable of detecting antonyms more effectively than other existing embeddings.

Word and Text Similarity Task. As a tool of antonym detection, it is worth celebrating that our WE-TSD method performs well on the GRE antonym detection task. However, it is far from enough as a general word embedding. In order to demonstrate the efficacy and task adaptability of our methods, we then conduct experiments on a basic task, word similarity evaluation.

Table 1. Results on the GRE antonym detection task. The best values are marked in bold font.

Model	DevSet			TestSet(950)			TestSet(790)		
	Prec.	Rec.	F	Prec.	Rec.	F	Prec.	Rec.	F
Encatra lookup	0.65	0.61	0.63	0.61	0.56	0.59	-	-	-
WordNet and Roget lookup	1.00	0.49	0.66	0.98	0.45	0.62	0.98	0.45	0.61
WE-T Model	0.92	0.71	0.80	0.90	0.72	0.80	0.90	0.72	0.80
WE-D Model	0.09	0.08	0.09	0.08	0.07	0.07	0.07	0.07	0.07
EnCarta PILSA	0.88	0.87	0.87	0.81	0.80	0.81	-	-	-
WordNet & Roget BPTF	0.88	0.88	0.88	0.82	0.82	0.82	-	-	-
WE-TD Model	0.92	0.91	0.91	0.90	0.88	0.89	0.89	0.87	0.88
WE-TM Model	0.92	0.91	0.91	0.91	0.89	0.90	0.89	0.87	0.88
WE-S Model	0.88	0.87	0.87	0.83	0.81	0.82	0.81	0.79	0.80
WE-TSD Model	0.95	0.92	**0.935**	0.93	0.91	**0.92**	0.92	0.90	**0.91**

In word similarity experiment, we compare our model with many baselines such as path similarity based on hypernym-hyponym structure in WordNet by [16], mutual information based on WordNet by [15], Word2Vec and LDA similarity proposed by [9] based on English Wikipedia data, WebJaccard algorithm based on Google Search webpages, WE-TD by [14] and so on. Our experimental results are listed in Table 2. The results of our model **WE-TSD** surpass both the **Gensim Word2Vec** and **WE-TD** model on WS353(Sim), WS353(Combined) and MEN-TR-3k dataset. Since our embedding aims to reflect the semantic similarity rather than the relatedness between words, **WE-TSD** performs not that well on WS353(REL). On WS353(REL) dataset, our model is slightly inferior to the **Gensim Word2Vec** model, but the results are still acceptable and outperform the other methods, which shows the effectiveness and versatility of our model.

In text similarity experiment, we compare our word embeddings with WE-TD Model, Gensim Word2Vec Model and GLOVE Word Representation Model to

Table 2. Spearman's rank correlation coefficients in different models. The best values are marked in bold font.

Model	WS353(Sim)	WS353(Rel)	WS353(Com)	MEN-TR-3k
Path Similarity on WordNet	0.347	0.262	0.315	0.298
Mutual Information on WordNet	0.388	0.257	0.349	0.325
Gensim Word2Vec on Wiki	0.651	**0.648**	0.650	0.611
LDA Method on Wiki	0.660	0.487	0.575	0.614
WebJaccard on Google Search	0.277	0.050	0.157	0.105
WE-TD Model on Wiki	0.621	0.617	0.620	0.717
WE-TM Model on Wiki	0.625	0.615	0.621	0.720
WE-S Model on Wiki	0.669	0.635	0.650	0.727
WE-TSD Model on Wiki	**0.675**	0.635	**0.656**	**0.732**

Table 3. The calculation of text similarity using CNN. The best values are marked in bold font.

Model	Test set	Validation set
GLOVE Word Representation	0.790	0.832
Gensim Word2Vec	0.794	0.831
WE-TD Model	0.715	0.733
WE-TM Model	0.725	0.746
WE-S Model	0.784	0.831
WE-TSD Model	**0.808**	**0.859**

demonstrate the applicability and efficacy of our antonym-sensitive embeddings on this task. The experimental results are listed in Table 3. Our WE-TSD model outperforms the other embeddings on both test set and validation set.

As is shown above, our model consists of three important parts, namely Modified Skip-Gram, Thesauri based and SentiWordNet based model. In fact, all of them are necessary. The first part is the most basic one and it assures that all the existing word in Wiki Corpus are taken into account in our model. The

second part shows its strength in antonym detection task, and the third part plays a vital role in word and task similarity tasks. Our model performs well on all of the three tasks, demonstrating its effectiveness and task adaptability.

5 Conclusions

In this paper, we propose a novel word embedding model to get better performance on discriminating antonyms from synonyms and our model achieves an F-score of 92% on GRE antonym detection task, outperforming the current state-of-the-art. Also this model has an satisfying performance on both word and textual similarity tasks, demonstrating its effectiveness and task adaptability. In future work, we plan to extend our ideas to train word embeddings which are capable of capturing other semantic relations, such as hyponyms and hypernyms.

References

1. Baccianella, S., Esuli, A., Sebastiani, F.: Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In: LREC, vol. 10, pp. 2200–2204 (2010)
2. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L.: Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. arXiv preprint arXiv:1708.00055 (2017)
3. Finkelstein, L., et al.: Placing search in context: the concept revisited. In: Proceedings of the 10th International Conference on World Wide Web, pp. 406–414. ACM (2001)
4. Harris, Z.S.: Distributional structure. Word **10**(2–3), 146–162 (1954)
5. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0026683
6. Kipfer, B.A.: Roget's 21st century thesaurus in dictionary form: the essential reference for home, school, or office. Laurel (1993)
7. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
8. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
9. Mikolov, T., Yih, W.T., Zweig, G.: Linguistic regularities in continuous space word representations. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 746–751 (2013)
10. Miller, G.A.: Wordnet: a lexical database for English. Commun. ACM **38**(11), 39–41 (1995)
11. Mohammad, S., Dorr, B., Hirst, G.: Computing word-pair antonymy. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 982–991. Association for Computational Linguistics (2008)
12. Mohammad, S.M., Dorr, B.J., Hirst, G., Turney, P.D.: Computing lexical contrast. Comput. Linguist. **39**(3), 555–590 (2013)

13. Nguyen, K.A., Walde, S.S.I., Vu, N.T.: Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction. arXiv preprint arXiv:1605.07766 (2016)
14. Ono, M., Miwa, M., Sasaki, Y.: Word embedding-based antonym detection using thesauri and distributional information. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 984–989 (2015)
15. Pantel, P., Lin, D.: Discovering word senses from text. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 613–619. ACM (2002)
16. Pedersen, T., Patwardhan, S., Michelizzi, J.: Wordnet: similarity: measuring the relatedness of concepts. In: Demonstration Papers at HLT-NAACL 2004, pp. 38–41. Association for Computational Linguistics (2004)
17. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533 (1986)
18. Shao, Y.: HCTI at semeval-2017 task 1: use convolutional neural network to evaluate semantic textual similarity. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pp. 130–133 (2017)