Improving Word Representations Using Paraphrase Dataset

53

Flávio Arthur O. Santos and Hendrik T. Macedo

Abstract

Recently, the NLP community has focused on finding methods for learning good vectorial word representations. These vectorial representations must be good enough to capture semantic relationships between words using simple vector arithmetic operations. Currently, two methods stand out: GloVe and word2vec. We argue that the proper usage of knowledge bases such as WordNet, Freebase and Paraphrase can improve even further the results of such methods. Although the attempt to incorporate information from knowledge bases in vectorial word representations is not new, results are not compared to that of GloVe nor word2vec. In this paper, we propose a method to incorporate the knowledge of Paraphrase knowledge base into GloVe. Results show that such incorporation improves GloVe's original results for at least three different benchmarks.

Keywords

GloVe · Paraphrase · Knowledge base · Word embeddings · Natural language processing

53.1 Introduction

Deep architectures of Multilayer Perceptron (MLP), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) are the state-of-the-art for many NLP tasks, such as automatic translation [22], question & answering [23], named-entity recognition [15], automatic text summarization [19] and sentiment analysis [16]. Most NLP solutions involving Deep Learning use word embeddings, which are vectorial word representations. Word embeddings are vectors of real numbers that represent a word; in this way, each word has its own word embedding. There are three main techniques for learning word embeddings:

1. Context-window based methods;
2. Semantic Relationship based methods;
3. Graph distance based methods.

All three methods have disadvantages in their development. Some of the methods in (1), such as [9] and [17], use only the local context of each word instead of the global context for training. The methods in (2) and (3) use the WordNet [18] and Freebase [3] knowledge bases to learn the word embeddings. The main disadvantage of the methods in (2) is that they use only a subpart of the aforementioned knowledge bases and do not consider the Paraphrase dataset [12], which contains a set of word pairs that are written differently but share the same meaning. Methods in (3) use the Leacock-Chodorow [6] distance in order to capture the semantic information between two words; not considering other distance measures is a limitation.

In this work, we used the Paraphrase knowledge base to enrich our training base of word embeddings and trained them using GloVe [20], which considers the global context of words. The hypothesis to be tested is that such combination improves vectorial word representations.

In Sect. 53.2 we present some related works. In Sect. 53.3 we describe the method used to improve the training base using the Paraphrase knowledge base. Section 53.4 details the experiments and discusses the results. Finally, we conclude the work in Sect. 53.5.

F. A. O. Santos · H. T. Macedo (✉)
Computer Science Postgraduate Program, Federal University of Sergipe, São Cristóvão, Brazil
e-mail: flavio.santos@dcomp.ufs.br; hendrik@dcomp.ufs.br

S. Latifi (ed.), *Information Technology – New Generations*, Advances in Intelligent Systems and Computing 738,
https://doi.org/10.1007/978-3-319-77028-4_53

53.2 Related Work

53.2.1 Context-Window Based Methods

Collobert et al. [9] implemented a *Neural Language Model* (NLM) where each vocabulary word i is related with a vector $v_i \; \varepsilon \; R^n$ of dimension n, the word embedding of i. A sentence s = $(s_1, s_2, s_{\cdot}.., s_l)$ of size l is represented for a vector x which is equal to concatenate vector of words embeddings from sentence s, $x = [v_{s_1} \, ; \, v_{s_2} \, ; \, \ldots ; \, v_{s_l}]$, $x \; \varepsilon \; R^{ln}$. After achieving x, it is propagated through a two layer neural network to obtain a score assign of how real this sentence is.

$$Score(x) = u^T(\sigma(Ax + b)) \tag{53.1}$$

A is an weight matrix such that $A \,\epsilon\, R^{h \times ln}$ and $b \,\epsilon\, R^h$ is the bias of first layer. The parameter h indicates how many units are in the layer f. $u^T \,\epsilon\, R^{1\times h}$ is the weight vector of output layer. The weight matrix and word embeddings of this model are trained using *Noise Contrastive Estimation* (NCE) [13], where, for each training sequence s, we build a noise sequence s_c. To build s_c we choose a word from s and replace it for a randomly selected word from vocabulary. Thus, we have a vector x for s and a vector x_c for s_c. To train a neural network able to achieve a high score on real sequences, we minimized the function cost at Eq. (53.2).

$$cost = max(0, 1 - Score(x) + Score(x_c)) \tag{53.2}$$

The word embeddings and parameters **A**, **b**, and **u** are trained with backpropagation using Stochastic Gradient Descent (SGD) over a training *corpus*.

Mikolov et al. [17] presents two architectures to learn word embeddings based on word context window inside an sentence, the Skip-gram and bag-of-words (CBOW). The skip-gram goal is: given a sentence s and a central word c of s, predict the context words of c. The CBOW goal is predict the central word c based on its context. Given an word sequence $w_1, w_2, w_3, \cdots, w_T$, the Skip-gram goal is maximize the E_{sk} function.

$$E_{sk} = \frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\geq c, j\neq 0} \mathbf{log} p(w_{t+j}|w_t), \tag{53.3}$$

where c is the context window size used. The most simple Skip-gram formula define $p(w_{t+j}|w_t)$ as:

$$p(w_{t+j}|w_t) = \frac{\mathbf{exp}({v'_{w_{t+1}}}^T v_{w_t})}{\sum_{n=1}^{N} \mathbf{exp}({v'_{w_n}}^T v_{w_t})} \tag{53.4}$$

53.2.2 Semantic Relationship Based Methods

There are knowledge bases that present semantic information about words, such as Freebase [3], WordNet [18], Dbpedia [2], NELL [7]. Often, knowledge is represented by $t = (w_i, r, w_j)$, where r indicate an semantic relationship between words w_i e w_j. Some models, e.g. TransE [4], Neural Tensor Network [21], try to learn word representations from this semantic information: tuple t as input and the output is a score indicating how real is the relationship r between words w_i e w_j.

53.2.3 Graph Distance-Based Methods

Fried and Duh [11] proposes the *Graph Distance* (GD) model. The goal of GD is to train the words embeddings such that its similarity is equal to LCH distance between the respective words in WordNet database. Its objective function is:

$$L_{GD}(v_i, v_j) = (\frac{v_i v_j}{||v_i||^2||v_j||^2} - [a \times LCH(w_i, w_j) + b])^2, \tag{53.5}$$

where v_i and v_j are word embeddings of w_i e w_j words, respectively. The GD uses the parameters **a** and **b** to put the LCH distance in the same scale as cosine similarity between v_i e v_j

53.3 Model

Paraphrase is the task of rewrite an sentence p using different words, but keeping the meaning of p. Word level Paraphrase is when we rewrite an word w with different characters but keep the w word meaning. Ganitkevitch et al. [12] presents an database for Paraphrase (PPDB). This database has around 73 million paraphrases at sentence level and 8 million paraphrases at word level. The PPDB is divided into six sizes: S, M, L, XL, XXL, XXXL, in crescent order. The subpart S, minor part, has a higher precision score. In this work, we selected the PPDB subpart S and used its 473 thousand word level paraphrases. We implemented the $getparaphrase(word = w)$ method which uses S; it randomly selects one paraphrase of the word w.

In this work, we used the GloVe [20] model to train the word embeddings. It is a context window based method. It uses the word global context of training corpus. Over its training, GloVe uses an word-to-word co-occurrence matrix X, where X_{ij} indicates how many times the word j is presented in word i context within the training corpus.

$$X_i = \sum_k X_{ik} \tag{53.6}$$

After building the matrix X, the GloVe's goal is to minimize J loss function:

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - log(X_{ij}))^2, \tag{53.7}$$

where V is the vocabulary size, w_i is the word embedding of central word i and $\tilde{w}_j$ is the word embedding of the context word j. Thus, we have two word embedding matrices, W and $\tilde{W}$. Equation (53.8) defines f(x) function.

$$f(x) = \begin{cases} (x/x_{max})^{\alpha}, & \text{if} x < x_{max} \\ 1, & \text{otherwise} \end{cases} \tag{53.8}$$

At GloVe original work, the authors use $\alpha = 3/4$ and $x_{max} = 100$ to train the word embeddings and perform the experiments.

```
input : matrix X, int V
for $i \leftarrow 1$ to $V$ do
    $p = getparaphrase(i)$;
    for $j \leftarrow 1$ to $V$ do
        if $X_{ij} == 0 \, and \, X_{pj} ! = 0$ then
            $X_{ij} := X_{pj}$
        end
    end
end
```

Algorithm 1: Algorithm used to enhance the X matrix

We use Algorithm 1 to enhance our X co-occurrence matrix and perform the GloVe's training. For each vocabulary word v, it randomly selects an paraphrase x and uses its context to fill the empty slots of v. In other words, it appends more information in our X matrix. This idea is valid because the words x and v have the same meaning, so the word v can be placed at the contexts of the word x.

53.4 Experiments

53.4.1 Evaluation Methods

We use three different benchmarks to evaluate the word embedding: (1) SimLex999 [14], (2) MEN [5] and (3) WordSimilarity-353 (WS353) [1]. They all measure the semantic similarity between two words. Each dataset has a set of tuples $t = (word1, word2)$, where each tuple has a score indicating how word1 and word2 are semantically related. This score is defined by arithmetic mean of a score set defined by humans.

The great difference between the SimLex999 and the other two, is that it explicitly evaluates the semantic similarity between two words, whereas the MEN and WS353 also consider the relatedness between two words. For instance, the tuple (Freud, Psychology) has a low score in SimLex999 but has a high score in MEN and WS, since the name Freud has a high relation with psychology.

For each dataset, we compute the cosine similarity between the word embeddings of word1 and word2 of its tuple $t = (word1, word2)$, so we have the semantic similarity of its word embeddings. In the end, we calculate the Spearman correlation between word embeddings semantic similarity and humans semantic similarity to obtain how good is our word embeddings on that dataset.

53.4.2 Corpora and Training Details

To perform the experiment, we use the 1 Billion Word Language Model Benchmark [8] corpus. This corpus has approximately 1 billion tokens. We tokenize and lowercase every word in the corpus using the Stanford tokenizer. We build a vocabulary with the most 100 thousand frequent words and produce the X co-occurrence matrix. To build the X matrix, we use an context window of size 10.

For every experiment, we use a $x_{max} = 100$, $\alpha = 0.75$ and train the GloVe model using Adagrad [10] and stochastically select elements with values different of zero from X. The initial learning rate was 0.05. We execute 100 training iterations of each word embedding for every experiment. In this work, we use W + W̃ as our final word embedding matrix. We use the GloVe original implementation to train our word embeddings and keep its default settings.

53.4.3 Results

In Table 53.1 we present the best achieved accuracy values. It can be observed that using the Paraphrase dataset to improve GloVe's co-occurrence matrix X presents an improvement in every scenario. Another important aspect to be noted is the word embeddings' dimension: experiments with bigger word embeddings also present better results.

Table 53.1 Best results for each model

Model	Size	SimLex999	WS353	MEN
GloVe	100	21.85	25.95	44.00
GloVe-P	100	21.89	27.75	44.85
GloVe	200	22.74	27.60	46.87
GloVe-P	200	23.66	27.93	47.92

P paraphrase

Fig. 53.2 SimLex999 results

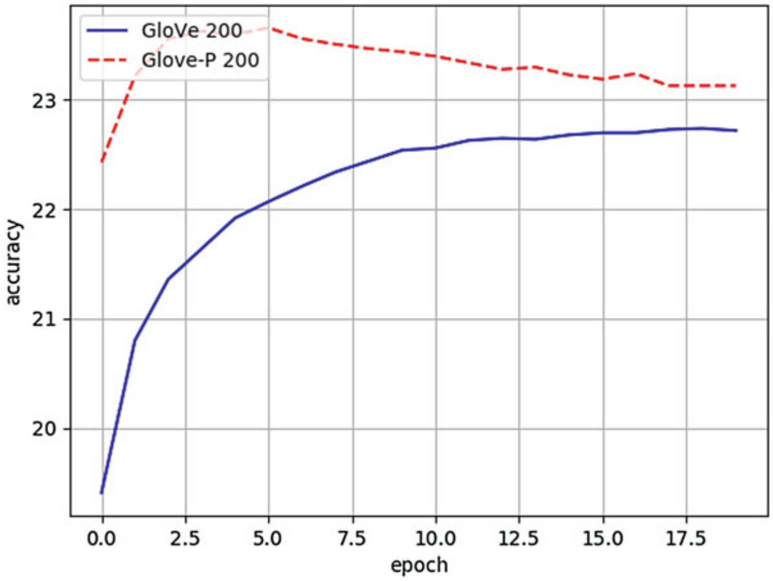
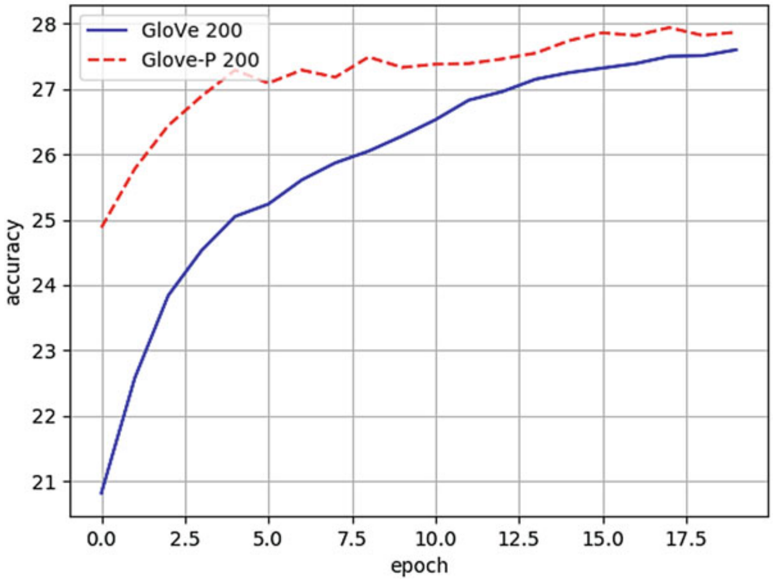


Fig. 53.3 MEN results



Figures 53.2, 53.3, and 53.4 present the accuracy evolution for Glove 200 and Glove-P 200 on the following benchmarks: SimLex999, MEN, and WS353, respectively. Aside from the fact that Glove-P 200 presents better results in every epoch and every evaluation, it is clear that during the first epochs, Glove-P shows a better improvement when compared to Glove 200. This is important due to the fact that high computational power is not always available to enable long-run training sessions.

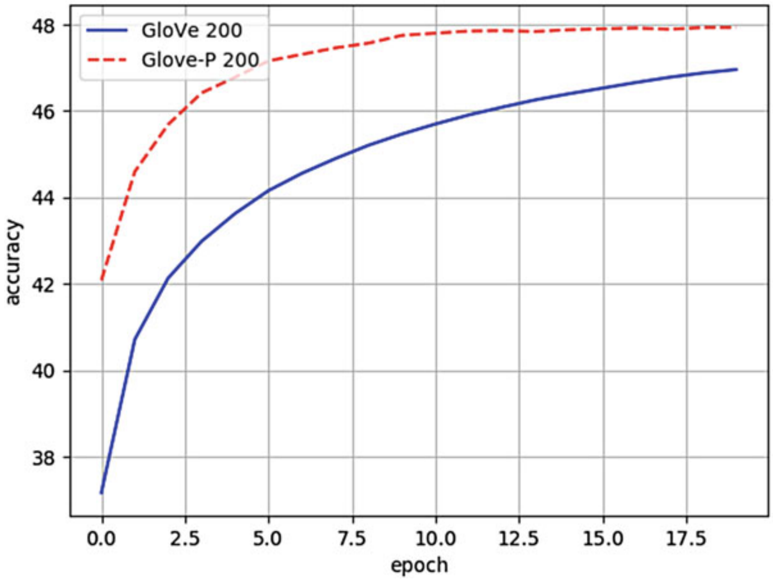
53.5 Conclusion

Vectorial word representations are important for obtaining good results in NLP tasks using machine learning algorithms. Recently, some works have tried to incorporate information from knowledge bases in order to improve the learning of word embeddings. However, these works have not tried to achieve the state-of-the-art results in their methods. Also, they usually limit their experiments with the use of self-tailored datasets.

In this work, we have proposed a modification on the GloVe method, the state-of-the-art in word representation benchmarks. In particular, we presented a method to incorporate the knowledge of the Paraphrase dataset into GloVe's co-occurrence matrix. We have used an universal dataset to train the word embeddings and results have shown improved word embeddings if compared to GloVe" original approach.

As future work, we intend to come up with a method to incorporate similar knowledge into other relevant learning methods such as word2vec.

Fig. 53.4 WS353 results



Acknowledgements The authors thank CAPES and FAPITEC-SE for the financial support [Edital CAPES/FAPITEC/SE No 11/2016 - PROEF, Processo 88887.160994/2017-00] and LCAD-UFS for providing a cluster for the execution of the experiments. The authors also thank FAPITEC-SE for granting a graduate scholarship to Flávio Santos, CNPq for granting an productivity scholarship to Hendrik Macedo [DT-II, Processo 310446/2014-7].

References

1. E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca, A. Soroa, A study on similarity and relatedness using distributional and wordnet-based approaches, in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics (2009), pp. 19–27
2. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, DBpedia: a nucleus for a web of open data, in *The semantic web* (Springer, Berlin, 2007), pp. 722–735
3. K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data* (ACM, New York, 2008), pp. 1247–1250
4. A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in *Advances in Neural Information Processing Systems* (2013), pp. 2787–2795
5. E. Bruni, N.-K. Tran, M. Baroni, Multimodal distributional semantics. J. Artif. Intell. Res. **49**(2014), 1–47 (2014)
6. A. Budanitsky, G. Hirst, Semantic distance in wordnet: an experimental, application-oriented evaluation of five measures, in *Workshop on WordNet and Other Lexical Resources*, vol. 2 (2001), p. 2
7. A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr., T.M. Mitchell, Toward an architecture for never-ending language learning, in *AAAI*, vol. 5 (2010), p. 3
8. C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, T. Robinson, One billion word benchmark for measuring progress in statistical language modeling (2013, preprint). arXiv:1312.3005
9. R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa, Natural language processing (almost) from scratch. J. Mach. Learn. Res. **12**, 2493–2537 (2011)
10. J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**, 2121–2159 (2011)
11. D. Fried, K. Duh, Incorporating both distributional and relational semantics in word representations (2014, preprint). arXiv:1412.4369
12. J. Ganitkevitch, B. Van Durme, C. Callison-Burch, PPDB: the paraphrase database, in *Proceedings of NAACL-HLT*, Atlanta, GA, Association for Computational Linguistics (2013), pp. 758–764
13. M. Gutmann, A. Hyvärinen, Noise-contrastive estimation: a new estimation principle for unnormalized statistical models, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), pp. 297–304
14. F. Hill, R. Reichart, A. Korhonen, Simlex-999: evaluating semantic models with (genuine) similarity estimation. Comput. Linguist. **41**, 665–695 (2016)
15. C. AEM Júnior, L.A. Barbosa, H.T. Macedo, S.E. Súo Cristóvão, Uma arquitetura híbrida lstm-cnn para reconhecimento de entidades nomeadas em textos naturais em língua portuguesa (2016)
16. H. Lakkaraju, R. Socher, C. Manning, Aspect specific sentiment analysis using hierarchical deep learning, in *NIPS Workshop on Deep Learning and Representation Learning* (2014)
17. T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in *Advances in Neural Information Processing Systems* (2013), pp. 3111–3119
18. G.A. Miller, Wordnet: a lexical database for english. Commun. ACM **38**(11): 39–41 (1995)
19. R. Paulus, C. Xiong, R. Socher, A deep reinforced model for abstractive summarization (2017, preprint). arXiv:1705.04304
20. J. Pennington, R. Socher, C.D. Manning, Glove: global vectors for word representation, in *EMNLP*, vol. 14 (2014), pp. 1532–1543
21. R. Socher, D. Chen, C.D. Manning, A. Ng, Reasoning with neural tensor networks for knowledge base completion, in *Advances in Neural Information Processing Systems* (2013), pp. 926–934,
22. Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., Google's neural machine translation system: bridging the gap between human and machine translation (2016, preprint). arXiv:1609.08144
23. C. Xiong, V. Zhong, R. Socher, Dynamic coattention networks for question answering (2016, preprint). arXiv:1611.01604