Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 157–160
DOI: 10.15439/2022F37
ISSN 2300-5963 ACSIS, Vol. 30

Extending Word2Vec with Domain-Specific Labels

Miloš Švaňa
VSB - Technical University of Ostrava
Department of Systems Engineering
17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czechia
Email: milos.svana@vsb.cz

***Abstract*—Choosing a proper representation of textual data is an important part of natural language processing. One option is using Word2Vec embeddings, i.e., dense vectors whose properties can to a degree capture the "meaning" of each word. One of the main disadvantages of Word2Vec is its inability to distinguish between antonyms. Motivated by this deficiency, this paper presents a Word2Vec extension for incorporating domain-specific labels. The goal is to improve the ability to differentiate between embeddings of words associated with different document labels or classes. This improvement is demonstrated on word embeddings derived from tweets related to a publicly traded company. Each tweet is given a label depending on whether its publication coincides with a stock price increase or decrease. The extended Word2Vec model then takes this label into account. The user can also set the weight of this label in the embedding creation process. Experiment results show that increasing this weight leads to a gradual decrease in cosine similarity between embeddings of words associated with different labels. This decrease in similarity can be interpreted as an improvement of the ability to distinguish between these words.**

I. Introduction

TRANSFORMATION of text into a numerical representation is an important part of any natural language processing (NLP) problem. Considering the level of words, one can choose between alternatives ranging from simple one-hot encoding to complex language models such as ELMo [9], BERT [3], or GPT-3 [2]. Word2Vec embeddings lie in-between these two extremes in terms of level of complexity.

Word embeddings are dense vectors usually of several hundred dimensions with the ability to at least partly capture the meaning of each word. This meaning is based on each word's context, i.e., words that co-occur with a given target word. It is captured by the relative position of different words in the embedding vector space. Words with similar meaning should be represented by vectors close to each other, while dissimilar words should be more distant. Moreover, basic operations such as addition or subtraction enable the derivation of representations for new words. A common example is the subtraction of the embedding for the word *man* from of the word *king*, followed by the addition of the embedding of *woman*. the result should be close to the embedding of the word *queen*.

Originally proposed by [6], Word2Vec is an algorithm for creating word embeddings. It is based on a simple idea: words with similar meaning occur in similar contexts. This context is usually defined by surrounding words. One issue with this line of thinking is that although Word2Vec works well for detecting synonyms, hyponyms or hypernyms [5, 14], it can't easily distinguish between two antonyms [11, 1]. Hence, words such as *good* and *bad* are often represented by relatively similar embeddings.

Several authors, including [8] or [4], addressed this issue by extending the basic Word2Vec algorithm to consider not only the context of words, but also thesauri information. In cited papers, several well known public datasets such as WordNet or Roget are used. [10] claim that the information about antonyms can be extracted from the geometry of the embedding space with a method called *contrasting maps*.

This paper describes a simple modification of the Word2Vec algorithm for considering domain specific document labels data during embedding creation. In contrast with the aforementioned work, presented approach is more flexible and can be adopted to many domains of interest.

In the paper, the modification is applied in the domain of finance. The problem can be stated as follows: We have set of tweets related to a specific publicly traded company. It is assumed that certain words occur more frequently when the stock price of a company drops, while others are used more when the stock price rises. In many financial analysis tasks it would be useful to represent words with occurrences associated with these opposite situations with vectors that are distant from each other. Can we improve on the basic Word2Vec algorithm by considering the information about stock price increase or decrease in the time of tweet publication?

Incorporating this information directly into word embeddings could be helpful in risk or return prediction. Some researchers, e.g. [12] or [13] are already using the basic Word2Vec model for similar tasks and this work could improve the prediction power of their models.

The remainder of this paper is organized as follows: Section II provides a basic overview of the Word2Vec algorithm, section III describes a modification for considering domain-specific labels during embedding creation, and section IV introduces the experiments performed to evaluate this modification. Results of these experiments are then presented and discussed in section V. Finally, the presented work is summarized in section VI. This section also discusses future research opportunities.

II. Word2Vec Overview

The Word2Vec algorithm is based on a simple feed-forward neural network with a single hidden layer. The number of

neurons in this hidden layer determines the dimensionality of the embedding space. Word embeddings are found by training this neural network in one of two ways – **continuous bag of words** (CBoW) and **skip-gram**.

The extension presented in this paper is based on the skip-gram model. When using this approach, the neural network uses the one-hot encoded target word as it's input and tries to predict it's multi-hot encoded context. The objective of the skip-gram model can be further formulated as maximizing the log probability [7]:

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

where T is the total number of target words, and c is the context window size, i.e., the number of words before or after the target word in the text to consider as skip-gram output.

The training sets for both CBoW and skip-gram methods are practically identical and can be created easily from a corpora of text documents. After the neural network is trained, the embeddings of different words are simply the weights of connections between the input element representing a given word in one-hot encoding and all neurons of the hidden layer.

In their subsequent paper, [7] extended the Word2Vec algorithm to improve both its performance in terms of training time and its accuracy. Two modifications were proposed:

- **Frequent word subsampling**: Words that occur frequently in the text are omitted from the document with probability:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{2}$$

  where $f(w_i)$ is the frequency of the word w_i and t is a threshold usually around 10^{-5}. Most frequent words include articles such as *the*, *a* and *an* that do not carry that much information about the contextual meaning of a specific word.
- **Negative sampling**: Leads to a modified objective function:

$$\log\,\sigma({v'_{w_O}}^T v_{w_I}) + \\ + \sum_{i=1}^{k}\mathbb{E}_{w_i\sim P_n(w)}\left[\log\,\sigma(-{v'_{w_i}}^T v_{w_I})\right] \tag{3}$$

  which in practical terms causes only k words that are missing in a given context to update their embeddings during training.

Experimental results show that these extensions lead to both more efficient training, and to better quality of final embeddings. Proposed extension implements word subsampling, but not negative sampling.

III. Proposed Extension

As discussed in introduction, one of Word2Vec's disadvantages is antonym representation. Antonyms such as *good* and *bad* are often surrounded by similar words, hence their Word2Vec embeddings are relatively similar.

I propose an extension of the Word2Vec model that allows for consideration of domain specific document labels to better distinguish between words related to different classes. In contrast to previous work presented in section I, it does not rely on general purpose thesauri.

The extension is based on the idea of modifying the output to be predicted by the neural network. In addition to context (surrounding words) for a certain input word, the neural network also has to predict the document class.

This extension further lets the user set the weight of this class label prediction relative to word context prediction. Modified objective function can then be formulated as:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}(\log\, p(w_{t+j}|w_t) + u\,\log\, p(d_t|w_t)) \tag{4}$$

where u is the label prediction weight and d_t is the document label associated with word w_t. This weight is implemented as replication of the output neurons for label prediction u times.

IV. Experiment design

To verify that the extension helps the Word2Vec algorithm better consider domain specific labels, an experiment with the goal of creating embeddings from tweets related to a publicly traded company was performed. Each tweet was labeled with a label "1" if the stock price increased one day after tweet publication. If the stock price decreases, the tweet was labeled with "0". Difference of 2 subsequent daily close prices was used to determine the label. The basic skip-gram model was then extended to predict not only the context of a word but also the label representing the stock price increase or decrease.

The hypothesis to test can be stated as follows: When the price change label is considered during embedding creation, the distance between embeddings of words occurring more frequently on price decrease and embeddings of words occurring more frequently on price increase will be greater as compared to the embeddings created with the standard Word2Vec skip-gram model. Moreover, this distance should grow with the weight of this price change label represented by u in equation 4.

A. Data and Preprocessing

The experiment utilized tweets related to the Walt Disney Company, which is publicly traded on the New York Stock Exchange under the *DIS* symbol. Tweets published between January 1, 2017 and December 31, 2020 were used to train the Word2Vec model. The *$DIS* "cashtag" was used to find tweets related to the company. 45 000 tweets containing 401 000 words were further randomly selected in order to reduce both the training time and memory use.

These tweets were preprocessed before they were used as input for the skip-gram model. Following steps were performed:

- the text was transformed to lower-case,
- cashtag symbols *$*, hashtag symbols *#* and the user mention symbols @ were removed,

- all other non-alphanumic symbols were removed,
- all HTML tags were removed,
- all URLs were replaced by a "__URL__" placeholder,
- and all numbers were replaced by a "__NUMBER__" placeholder.

B. Hypothesis Evaluation

To evaluate the aforementioned hypothesis, the polarity of each word was calculated as the difference between the number of occurrences of a given word in tweets associated with a price increase (positive occurrences) and in tweets associated with a price decrease (negative occurrences).

Since the dataset contains a different number of positive and negative tweets, the polarity of negative tweets was further modified using the following equation:

$$pol_m(w) = \frac{N_{pos}}{N_{neg}}\ pol(w) \tag{5}$$

where $pol_m(w)$ is the modified polarity of word w, $pol(w)$ is the basic polarity of the same word calculated as the difference between positive and negative occurrences, and N_{pos} and N_{neg} is the total number of tweets related to price increase and price decrease respectively.

Top 75 positive words were then paired with top 75 negative words (most negative word was paired with the most positive word, 2nd most positive word was paired with 2nd most negative word and so on). Then the distances between the embeddings of paired words were examined.

Furthermore, a list of 10 specific antonym pairs was constructed. This list was then verified to make sure that one word in the pair has indeed negative polarity, while the other word's polarity is positive. These antonym pairs are listed in table IV-B. Distances between the embeddings were again examined.

To consider different vector norms, cosine similarity was used as a measure of distance. According to Agudo [1], cosine similarity is also preferred for synonym or antonym detection tasks.

All experiments were implemented in Python 3.9 using well-known libraries including *numpy*, *pandas* and *Keras*.

Positive word	Negative word
buying	selling
upgraded	downgraded
raised	lowered
strength	weakness
ahead	delayed
bullish	bearish
up	down
above	below
high	low
positive	negative

TABLE I
PAIRS OF ANTONYMS WHOSE DISTANCES WERE EXAMINED DURING EXPERIMENTS

V. Experiment Results

Table II shows the cosine similarity measure for the antonym pairs listed above. These results include three different levels of label weight, as well as similarity derived from the embeddings trained by the well-known Gensim[1] Word2Vec implementation. With label weight set to 1, the average cosine similarity is very close to the similarity exhibited by Gensim embeddings. This small difference can be attributed to the random nature of the neural network training process. However, as the label weight increases the similarity between embeddings starts to drop significantly. When the label weight is set to 100 the embeddings become almost orthogonal.

These results are confirmed by the second test examining 75 pairs of top positive and top negative words. Noteworthy negative words include "down", "coronavirus", "below", "downgraded" or "risk". Interesting positive words include "higher", "nice", "streaming", "up" or "nflx". Using the same label weight values as in table II, mean cosine similarities across all word pairs were 0.36 ($u = 1$), 0.315 ($u = 10$) and 0.046 ($u = 100$). The cosine similarity mean for Gensim embeddings was 0.346. These results manifest the same behavior as the results for 10 selected antonym pairs.

Both experiments support the hypothesis stated in section IV. Given a specific minimum weight, domain-specific labels can indeed help increase the distance between relevant word embeddings. Moreover, this distance increment grows with the label weight.

VI. Conclusion

This paper explored the possible utilization of domain-specific labels during word embedding creation with the Word2Vec algorithm. An extension to the skip-gram model was proposed and evaluated in an experiment where word embeddings were created from a dataset of tweets related to the Walt Disney company. Results of this experiment show that the extension helps distinguish between words whose occurrence is correlated with different labels (in this case stock price increase and decrease). Furthermore, the cosine similarity between such words decreases as the label weight increases.

Presented experiments were performed on a relatively small dataset of 45000 tweets related to a single company. The proposed extension should therefore be examined on significantly larger amounts of data in the future. Moreover, the extension implementation used during experiments is not ready to be deployed to production. Further performance improvements are needed. Combining the extension with negative sampling should also be examined.

The extension was compared with a standard Word2Vec implementation provided by the Gensim library. Additional comparison to the models proposed by [8] or [4] would be beneficial. One of the potential benefits of the presented extension is its ability to consider any domain-specific labels instead of relying on a specific thesaurus.

The work presented in this paper is a part of a larger project with the aim of examining if sentiment and other information

[1] https://radimrehurek.com/gensim/

TABLE II
COSINE SIMILARITY BETWEEN WORD EMBEDDINGS COMPARED ACROSS VARIOUS CONFIGURATIONS

Positive word	Negative word	Gensim	u = 1	u = 10	u = 100
buying	selling	0.772	0.753	0.156	0.009
upgraded	downgraded	0.961	0.959	0.579	0.082
raised	lowered	0.942	0.937	0.875	0.044
strength	weakness	0.911	0.909	0.594	0.022
ahead	delayed	0.461	0.461	0.547	0.023
bullish	bearish	0.912	0.919	0.151	0.008
up	down	0.602	0.640	0.154	0.004
above	below	0.769	0.735	0.059	0.004
high	low	0.876	0.869	0.117	0.008
positive	negative	0.840	0.844	0.392	0.014
Mean		**0.804**	**0.803**	**0.362**	**0.022**

extracted from social media can be used to improve mean-risk investment portfolio optimization models. In the future I plan to examine the usefulness of the presented extension for stock price and risk prediction and compare it with complex language models such as BERT.

ACKNOWLEDGMENT

This paper was supported by the SGS project No. SP2022/113. This support is gratefully acknowledged.

REFERENCES

[1] M. G. Agudo. An analysis of word embedding spaces and regularities. 2019.

[2] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, and T. Henighan. Language models are few-shot learners. page 25, 2020.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1423. URL http://aclweb.org/anthology/N19-1423.

[4] Z. Dou, W. Wei, and X. Wan. Improving word embeddings for antonym detection using thesauri and SentiWordNet. In M. Zhang, V. Ng, D. Zhao, S. Li, and H. Zan, editors, *Natural Language Processing and Chinese Computing*, volume 11109, pages 67–79. Springer International Publishing, 2018. ISBN 978-3-319-99500-7 978-3-319-99501-4. doi: 10.1007/978-3-319-99501-4_6. URL http://link.springer.com/10.1007/978-3-319-99501-4_6. Series Title: Lecture Notes in Computer Science.

[5] A. Handler. An empirical study of semantic similarity in WordNet and word2vec. page 23, 2014.

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. 2013. URL http://arxiv.org/abs/1301.3781.

[7] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. 2013. URL http://arxiv.org/abs/1310.4546.

[8] M. Ono, M. Miwa, and Y. Sasaki. 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*, pages 984–989. Association for Computational Linguistics, 2015. doi: 10.3115/v1/N15-1100. URL http://aclweb.org/anthology/N15-1100.

[9] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. 2018. URL http://arxiv.org/abs/1802.05365.

[10] I. Samenko, A. Tikhonov, and I. P. Yamshchikov. Intuitive contrasting map for antonym embeddings. 2021. URL http://arxiv.org/abs/2004.12835.

[11] Y. Shao, S. Taylor, N. Marshall, C. Morioka, and Q. Zeng-Treitler. Clinical text classification with word embedding features vs. bag-of-words features. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2874–2878. IEEE, 2018. ISBN 978-1-5386-5035-6. doi: 10.1109/BigData.2018.8622345. URL https://ieeexplore.ieee.org/document/8622345/.

[12] M. R. Vargas, B. S. L. P. de Lima, and A. G. Evsukoff. Deep learning for stock market prediction from financial news articles. In *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pages 60–65. IEEE, 2017. ISBN 978-1-5090-4253-1. doi: 10.1109/CIVEMSA.2017.7995302. URL http://ieeexplore.ieee.org/document/7995302/.

[13] H.-Y. Yeh, Y.-C. Yeh, and D.-B. Shen. Word vector models approach to text regression of financial risk prediction. 12(1):89, 2020. ISSN 2073-8994. doi: 10.3390/sym12010089. URL https://www.mdpi.com/2073-8994/12/1/89.

[14] L. Zhang, J. Li, and C. Wang. Automatic synonym extraction using word2vec and spectral clustering. In *2017 36th Chinese Control Conference (CCC)*, pages 5629–5632. IEEE, 2017. ISBN 978-988-15639-3-4. doi: 10.23919/ChiCC.2017.8028251. URL http://ieeexplore.ieee.org/document/8028251/.