Journal of Physics:
Conference Series

PURPOSE-LED
PUBLISHING™

PAPER • **OPEN ACCESS**

Continuous-bag-of-words and Skip-gram for word vector training and text classification

To cite this article: Haowen Xia 2023 *J. Phys.: Conf. Ser.* **2634** 012052

View the article online for updates and enhancements.

Continuous-bag-of-words and Skip-gram for word vector training and text classification

Haowen Xia

School of Computer Science and Engineering, Central South University, Changsha, 410083, China

8204203328@csu.edu.cn

Abstract. Natural language processing is one of the most challenging parts in the study of artificial intelligence and is widely used in real-life applications. One of the basic questions is how to calculate the probability of a particular text sequence appearing in a certain context. Word2Vec is a powerful tool that provides a solution to the question for its ability to transform words into word vectors, and to train in high efficiency on large datasets and corpora. It has many models of which Continuous-Bag-Of-Words and Skip-gram are of great significance and also known to many people. Furthermore, some extended techniques related to the models are also proposed in order to simultaneously decrease required training time and increase the rate of accuracy for the training. Even though there are now a number of papers that describe these fundamental concepts, the quality vary greatly. To better understand the models and their extensions, and how well they behave when used for real tasks, different combinations of the models and techniques are made in this paper so as to compare their performance in processing large input data and the ability for correct prediction in the task of text classification. This is done as it could lead to more provision of details and understandings of the model for subsequent researches on this field of study.

Keywords: natural language processing, bag of words, word2Vec, deep learning.

1. Introduction

Word embedding processes natural languages by representing words as vectors, which lays a solid groundwork for subsequent procedures in computers. Unlike one-hot encoding in which vectors consist of only zeros and ones, word vectors that are processed by word embedding consist of real-valued numbers that vary from one another. Digging deeper into the technique, it embeds words into a vector space in which words that are close in distance will also have similar meanings when they appear together in a text.

Word2Vec [1-3], which is proposed by Google in 2013, is one kind of technique of word embedding. Continuous-Bag-Of-Words (CBOW) and Skip-gram are two of its models which are widely researched and used by people over the years. CBOW aims to predict the word that is in the middle of a sentence using the surrounding words. Skip-gram, on the contrary, predicts surrounding words of a given word. There are also some extensions to the models for faster training and reduced computational costs when Word2Vec is confronted with excessively increased size of data. Of these extensions, Negative Sampling, Hierarchical Softmax and Subsampling of Frequent Words are three famous and significant ones that people are familiar with. More problems can be solved efficiently with the help of these models and extensions, as can be seen in [4-6].

Although there are currently many state-of-the-art methods and techniques that help better deal with issues and potential risks during the process of training [7,8], most of them are based on the classic models that are indispensable in the field. As a result, it is with great necessity that people learn about them when they are doing relevant researches.

Overall, the main contributions of this paper can be summarized as:

•The principles of both models, as well as their extensions, are recounted as prerequisites for experiments. The work is operated primarily by showing correlative ideas, formulas, and examples. Negative Sampling and Hierarchical Softmax are two important extended techniques that are discussed in this paper.

•A cross combination of these models and techniques are made and the efficiency of each combination are also compared through practical training. In the experiment, "text8" is used as the corpus for the training, and the experiments are all trained by gensim. In order to further compare the average training time as well as other information for each combination, another experiment is conducted for more scrupulous evaluation indicators of the combinations.

•A task of text classification is revised in order to see each combinations' performance in dealing with practical applications. The results could be a good evaluation for each combination.

2. Model architectures

Most papers use eradicate languages and mathematical symbols to recount the models, which often results in readers, especially non-expert researchers lacking of interest in proceeding to read. Therefore, in this paper, only prime words and a few crucial formulas are demonstrated in front of the readers while still maintaining their essence at the same time. At the very beginning, people often use one-hot encoding to represent words. However, when the size of the corpus is too large, it faces severe problems and takes unbearable amount of time for training. With the proposal of Continuous-Bag-Of-Words and Skip-gram, most of the problems related to that are now fixed. The two models are specific reflections of Word2Vec to represent words as vectors. In this section, the principles of these models and their extensions are discussed according to [9], as well as some comparisons based on models and extensions combined in different ways.

2.1. Continuous-bag-of-words architecture

This model is based on the thought that neighbouring words that appear closely in a text are expected to be highly similar in meaning, and words that appear far from each other are often dissimilar. Therefore, a centre word is expected to appear given the condition of appearance of neighbouring words. The probability P is expected as the below formula:

$$p = p(c|w_1)p(c|w_2) \cdots p(c|w_n) \tag{1}$$

to be as big as possible. In the above formula, c means the centre word, and w_i means words that surround the centre word. The model can be proved to perform well as a result of a big probability P. Going deeper into the theory, the goal is to obtain the maximum objective function by maximizing

$$obj = argmax \sum_{w \in text} \sum_{c \in context(w)} p(c|w;\theta) \tag{2}$$

in Formula (2), θ is a hyper-parameter.

2.2. Skip-gram

Contrary to the theory of CBOW, Skip-gram aims to predict the surrounding words of the centre word in a text. Mathematically speaking, the model optimizes the objective function by maximizing

$$\sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} log\, P\, (w_{t+j}|w_t) \tag{3}$$

Where c is the window size that is used for training. In this formula, $P(w_{t+j}|w_t)$ is usually defined using the Softmax function. More formally, Mikolov et al. (2013) uses $P(w_O|w_I)$ to represent the probability which is calculated by:

$$P(w_O|w_I) = \frac{exp(v\prime_{w_O}^{T} v_{w_I})}{\sum_{w=1}^{W} exp(v\prime_{w}^{T} v_{w_I})} \tag{4}$$

Note the related part in [1] by Mikolov that, a larger window size c will lead to more samples, and so it results in higher accuracy. However, it is found that it also results in longer training time, which is proportional to c.

3. Model techniques

Both CBOW, Skip-gram are excellent models that contribute to better processing word representations. But they also have trouble dealing with large size of input data, as a result, some techniques are proposed to solve the problem. The principles of Negative Sampling and Hierarchical Softmax are briefly discussed in this section. The goal of this paper is to make combinations of the models and techniques and compare their working efficiency.

3.1. Negative sampling

A new supervised-learning problem is bound to be created in the algorithm. Given a pair of words, the goal is to predict whether the pair fits the "context-target" structure. Let's say there is a sentence: "I would like a glass of black tea", it is easy to know that "black" and "tea" is a pair that fits the structure, it is therefore a positive sample, which will be labelled as 1. A negative sample, however, is a word randomly selected from the corpus to form a pair with one of the words from the positive sample. For example, if the word "king" is selected, then "black king" will be a negative sample, which will be labelled as 0. A supervised-learning problem is required in which the learning algorithm correctly predicts the target label using the input pair of words. Negative Sampling was defined by the objective function

$$log\, \sigma\, (v'^{T}_{w_O} v_{w_I}) + \textstyle\sum_{i=1}^{k} E_{w_i \sim P_n(w)}\, [log\, \sigma\, (-v'^{T}_{w_i} v_{w_I})] \tag{5}$$

that was put forward by Mikolov et al. (2013). In the formula, $P_n(w)$ is the noise distribution, which is calculated by the formula:

$$P_n(w) = \frac{U(w)^{\frac{3}{4}}}{Z} \tag{6}$$

Where Z is a normalization constant and $U(w)$ is a unigram distribution. The power is chosen based on experimental results. It is supposed that every positive sample will be used for training together with k negative samples, which can be concluded from Formula 6 that it is done over k negative samples from the noise distribution. In doing so, a small size of samples should be trained instead of training a larger one that might yield time costs.

3.2. Hierarchical softmax

An alternative that could also help decrease training time is Hierarchical SoftMax, which is an approximation based on binary tree. In the binary tree, each leaf node represents a label that corresponds to a word in the corpus, and is associated with a path starting from the root to the node, in which each non-leaf node corresponds to a vector, as can be seen in Figure 6. These non-leaf nodes also represent that binary classification decisions must be made. Instead of having to calculate the probability of one word, Only the probabilities of a sequence must be calculated, through which the normalization over all words at the expense of time is not required to be considered. The training speed of this technique is dependent on the structure of the binary tree, Huffman tree is chosen in this experiment as shorter paths are assigned to frequent words in the tree.

For a specific word, the conditional probability is:

$$p(w|Context(w)) = \prod_{j=2}^{l^w} p(d_j^w|\vec{x}_w, \theta_{j-1}^w) \quad (7)$$

Where p^w is the path from the root to the node w, l^w is the number of nodes contained in a path, $d_j^w \in \{0,1\}$ is the encoding of the jth node in path p^w , and $\theta_{j-1}^w \in R^m$ is the parameter vector in path p^w.

Many people have done researches on Hierarchical Softmax, Mikolov used Huffman tree as the basis of Hierarchical Softmax, by which our work is inspired. Mnih and Hinton also explored different structures of the tree to seek for one that works best for the model.

4. Experiments

4.1. Dataset

In this paper, two different datasets for the experiments are taken. “text8” are selected to be the corpus of the first experiment, and different sizes of the dataset are tested for the second experiment. 1MB, 10MB and 50MB of the corpus are in turn selected as input data to test the actual effect of each combination of models.

4.2. Independent task

CBOW and Skip-gram are basic models that are used for processing word vectors, Negative Sampling and Hierarchical Softmax are extended techniques to help training, which can result in higher accuracy and less training time.

In order to test their effect and make comparisons about it, across combination of these models and techniques are made, and their performance are also respectively tested. That is, four combinations, which are CBOW+Negative Sampling, CBOW+Hierarchical Softmax, Skip-gram+Negative Sampling and Skip-gram+Hierarchical Softmax are made, and each combination’s performance for processing the “text8” corpus are also tested. This is done in order to compare the models, as well as the techniques, and watch the phenomena that might occur during the process of training and give possible reasons for them.

Results 1: Vector dimensionality 100 and context size 5 are used as parameters for the model, each of the combination of techniques are tested and it is found that the Skip-gram model needs more training time than the CBOW model does, and Hierarchical Softmax also needs more training time than Negative Sampling does. Plus, the loss values of the Skip-gram model vary more greatly from one another than those of the CBOW model. It is easy to conclude that CBOW is more efficient than Skip-gram, and Negative Sampling is also more efficient than Hierarchical Softmax. The first conclusion could be attributed to the number of Softmax. In CBOW, there is only one Softmax for training, whereas in Skip-gram, there are k Softmax, where k is the number of context words. The second conclusion manifests the superiority of Negative Sampling in dealing with frequent words and lower dimensional vectors, along with the excellence of the Hierarchical Softmax in handling infrequent words. The results are shown in Table 1.

Table 1. Required training time of each combination. “hs” = Hierarchical Softmax; “ns”=Negative Sampling; “sg”=Skip-gram.

	CBOW+hs	CBOW+ns	sg+hs	sg+ns
time (s)	259.905	217.188	752.647	740.677

In the experiment, the loss values of each combination are also calculated. However, it is found that only the combination of CBOW+Negative Sampling has loss values that decrease after each epoch, which is shown in Figure 1. The loss values of other combinations will at first increase for a few epochs, and then decrease, which does not correspond to what is expected. One of the possible reasons is that, there are lots of inconsistencies within the model of Gensim itself, the rising-falling loss could be attributed to the side effects caused by those inconsistencies. More details are shown in Figure 2.

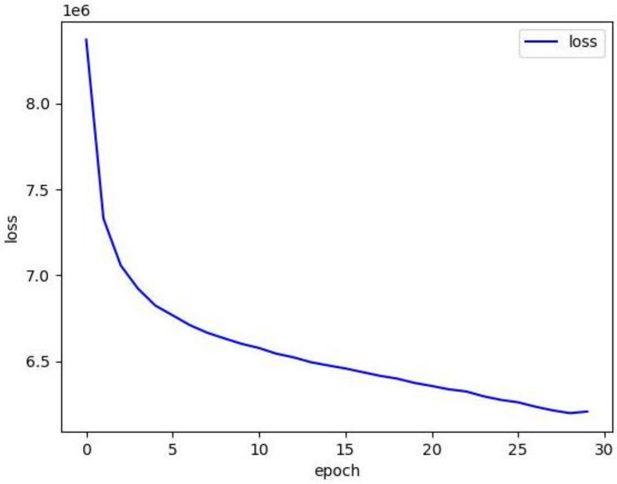


Figure 1. Curvilinear graph of loss values of CBOW+Negative sampling.

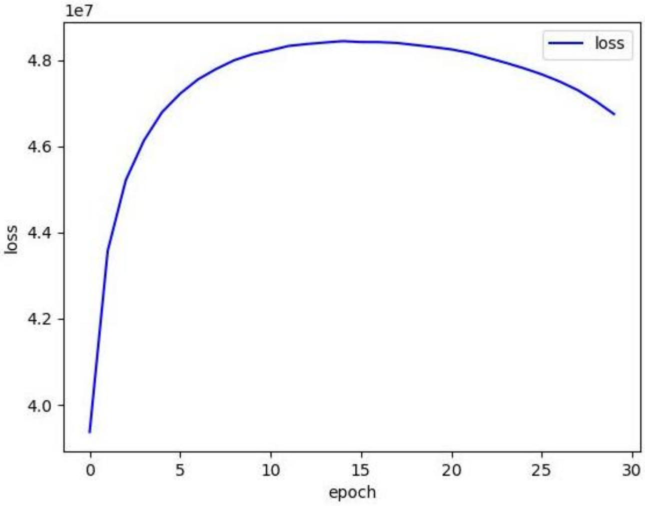


Figure 2. Curvilinear graph of loss values of CBOW+Hierarchical Softmax.

An intriguing point that is worth further discussion is also found. During the training process, it is noted that the loss value would become zero after a few epochs when adopting Hierarchical Softmax as the training technique no matter of the model that is initially chosen. At first, the author thought it was due to the incorrect setting of the learning rate; therefore, the author tested several different learning rates based on the CBOW model with Hierarchical Softmax adopted as the training technique to validate whether it was what the author had suspected. The results show that there are some changes to the loss values according to different learning rates, but are too weak to help support the hypothesis. The conclusion could be safely drawn that the loss values remain the same despite adjustments of learning rates, which illustrates that learning rate is probably not the reason why the loss value becomes zero after epochs. Considering the process, there could be a variety of reasons that help explain the phenomenon, it is believed that the model being overfitted might be the best among all the reasons to explain it.

4.3. Integrated task

Another experiment is conducted in order to obtain the average required training time of each combination of models and techniques, this is done because the possible effect of chance factors on the experiment also need to be eliminated. Each combination is tested a few times using different input data, the parameters of the model, the average time needed for training as well as the standard deviation are also calculated. The results are shown in Table 2-4.

Table 2. Results for 1MB of the corpus.

scale of data(MB)	compute_loss	sg	hs	mean_time	std_time
1	true	0	0	0.379	0.005
1	false	0	0	0.389	0.001
1	true	0	1	0.727	0.004
1	false	0	1	0.735	0.007
1	true	1	0	0.946	0.002
1	false	1	0	0.937	0.004
1	true	1	1	1.957	0.008
1	false	1	1	1.937	0.007

Table 3. Results for 10MB of the corpus.

scale of data(MB)	compute_loss	sg	hs	mean_time	std_time
10	true	0	0	3.845	0.035
10	false	0	0	3.946	0.032
10	true	0	1	7.810	0.053
10	false	0	1	7.841	0.014
10	true	1	0	10.727	0.030
10	false	1	0	10.765	0.011
10	true	1	1	23.109	0.023
10	false	1	1	23.224	0.082

Table 4. Results for 50MB of the corpus.

scale of data(MB)	compute_loss	sg	hs	mean_time	std_time
50	true	0	0	21.002	0.052
50	false	0	0	21.148	0.028
50	true	0	1	41.398	0.094
50	false	0	1	41.091	0.146
50	true	1	0	64.739	0.879
50	false	1	0	63.174	0.253

Table 4. (continued).

50	true	1	1	131.992	1.518
50	false	1	1	129.860	0.181

In order to better compare the performance of each combination, the multiples of the required training time for Skip-gram over CBOW are calculated, as well as Hierarchical Softmax over Negative Sampling, which are shown in Table 2 and Table 3. Obviously, the average training time increases with the growth of the scale of input data. It is found that the average training time of Skip-gram model is 2.5-3 times that of CBOW model, and is 1.9-2.1 times that of Negative Sampling when Hierarchical Softmax is adopted.

Table 5. Training time (of Skip-gram) / training time (of CBOW).

	1	2	3	4
1MB	2.49	2.41	2.69	2.64
10MB	2.79	2.73	2.96	2.96
50MB	3.08	2.99	3.19	3.16

Table 6. Training time (of Hierarchical Softmax) / training time (of Negative Sampling).

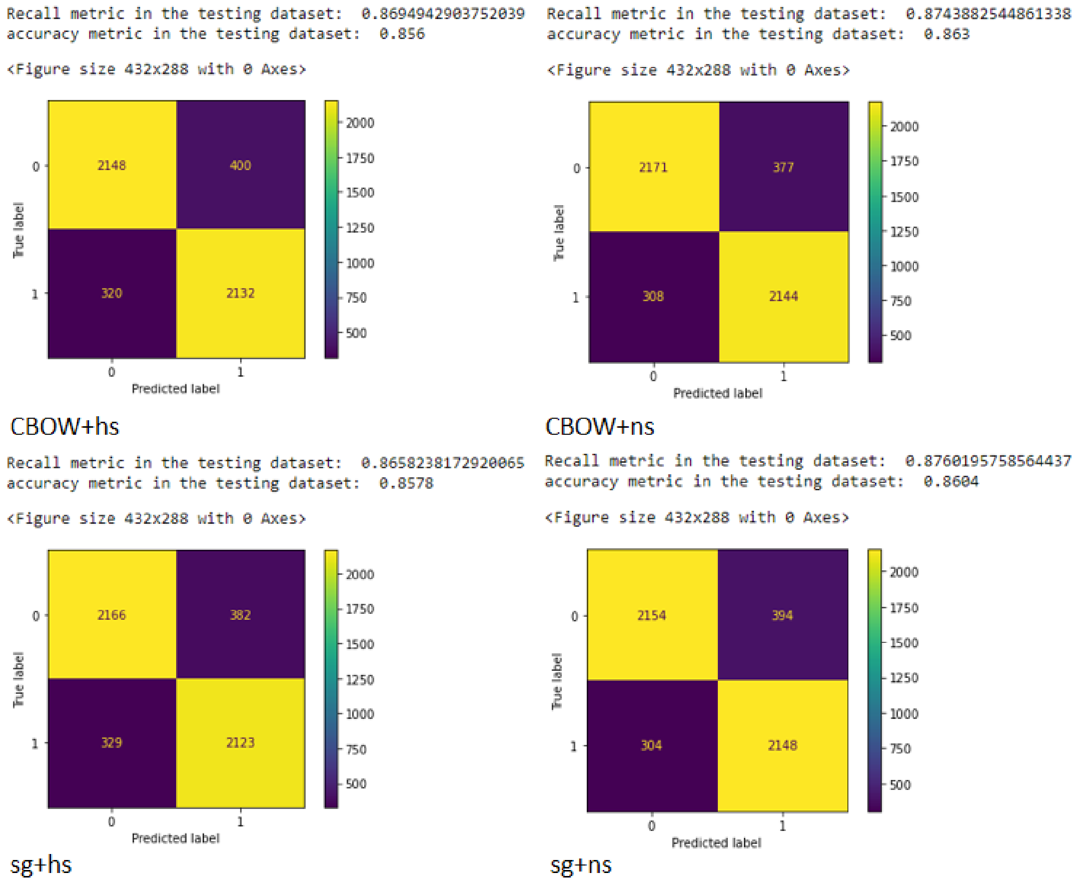
	1	2	3	4
1MB	2.07	2.07	1.92	1.89
10MB	2.15	2.16	2.03	1.99
50MB	2.04	2.06	1.97	1.94

4.4. Classification task

There are now many applications of Word2Vec, as can be seen in [10,11]. In this paper, A task of text classification is revised and a dataset that contains several film reviews are chosen for the task. Each film review in the dataset is labelled as either one or zero. The labels are used to identify if the review is a positive one or a negative one. They are grouped together as input data to the model so the performance of the models, which in detail, the ability to predict the right labels for each review can be detected.

Results2: Each combination is tested and the average accuracy metric as well as the recall metric are also calculated, which are shown in Figure 3. The results show that Skip-gram model shows no advantage over CBOW at the classification task, and Hierarchical Softmax is also approximately the same as Negative Sampling in the aspect of this task.

There are many types of tasks in which these models and techniques can be applied. Operating only one out of these tasks may seem insufficient in terms of support for their abilities, but the results shown in this paper can be provided as initial insight for those who carry out further research on this field of study, which is also one of the purposes of the author.



```
Recall metric in the testing dataset:  0.8694942903752039
accuracy metric in the testing dataset:  0.856

<Figure size 432x288 with 0 Axes>
```

CBOW+hs

```
Recall metric in the testing dataset:  0.8743882544861338
accuracy metric in the testing dataset:  0.863

<Figure size 432x288 with 0 Axes>
```

CBOW+ns

```
Recall metric in the testing dataset:  0.8658238172920065
accuracy metric in the testing dataset:  0.8578

<Figure size 432x288 with 0 Axes>
```

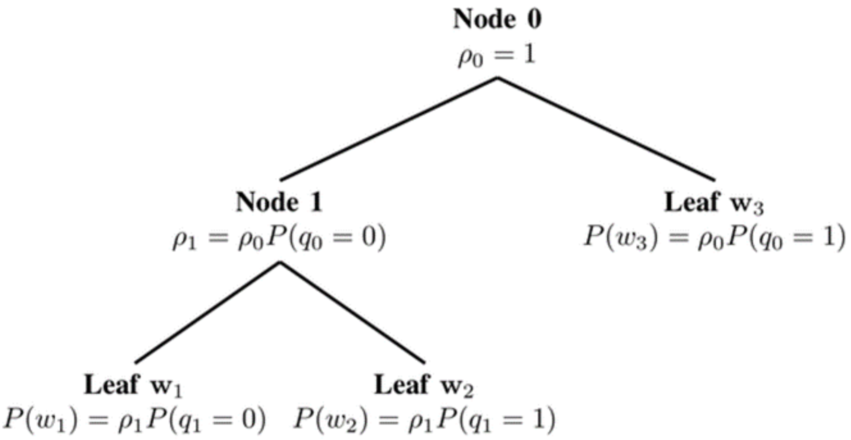
sg+hs

```
Recall metric in the testing dataset:  0.8760195758564437
accuracy metric in the testing dataset:  0.8604

<Figure size 432x288 with 0 Axes>
```

sg+ns

Figure 3. confusion matrices for the performance of each combination. (“hs” is Hierarchical Softmax, “ns” is Negative Sampling).



Node 0
$\rho_0 = 1$
Node 1
$\rho_1 = \rho_0 P(q_0 = 0)$
Leaf w_3
$P(w_3) = \rho_0 P(q_0 = 1)$
Leaf w_1
$P(w_1) = \rho_1 P(q_1 = 0)$
Leaf w_2
$P(w_2) = \rho_1 P(q_1 = 1)$

Figure 4. Structure of huffman tree [12].

5. Conclusion

In this paper, the basic concept of Word2Vec is introduced, and the principles of two important models of Word2Vec are discussed in detail. Besides, the concepts of some extended techniques for Word2Vec and their applications are also introduced. An experiment for testing the effect of the models and techniques is conducted and it is discovered that compared to Skip-gram, CBOW requires less training time and results in lower loss values. Plus, Negative Sampling seemingly performs better than Hierarchical Softmax but this difference depends on the frequency of a specific word. Generally, Negative Sampling is better at dealing with frequent words and lower dimensional vectors while Hierarchical Softmax works well for infrequent words. The loss values trained by these models and techniques range from bad to good, some of them do not even go along with what people think they should do; however, it should not be the only indicator for evaluating whether it is good training or not, some more indicators from different aspects should be utilized for evaluation. A task of text classification is conducted for comparing their performance in real applications. As can be seen from the confusion matrices, it is easy to observe the disparity among these combinations of models and extended techniques in terms of both recall and accuracy, and further, their predictive abilities.

Reference

[1] Mikolov, T., Sutskever, I., Chen, K., et al. (2013). Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 26.

[2] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[3] Sivakumar, S., Videla, L. S., Kumar, T. R., Nagaraj, J., Itnal, S., & Haritha, D. (2020, September). Review on Word2Vec Word Embedding Neural Net. In 2020 International Conference on Smart Electronics and Communication, 282-290.

[4] Yang, Z., Ding, M., Zhou, C., et al. (2020). Understanding negative sampling in graph representation learning. In Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, 1666-1676.

[5] Armandpour, M., Ding, P., Huang, J., & Hu, X. (2019). Robust Negative Sampling for Network Embedding. Proceedings of the AAAI Conference on Artificial Intelligence, 33(01), 3191-3198.

[6] Xu, K., Feng, Y., Huang, S., & Zhao, D. (2015). Semantic relation classification via convolutional neural networks with simple negative sampling. arXiv preprint arXiv:1506.07650.

[7] Ma, L., & Zhang, Y. (2015). Using Word2Vec to process big text data. In 2015 IEEE International Conference on Big Data, pp. 2895-2897.

[8] Lilleberg, J., Zhu, Y., & Zhang, Y. (2015). Support vector machines and word2vec for text classification with semantic features. In 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing, 136-140.

[9] Rong, X. (2014). word2vec parameter learning explained. arXiv preprint arXiv:1411.2738.

[10] Fulin, X., Yihao, D., & Xiaosheng, T. (2015). The architecture of word2vec and its applications. Journal of Nanjing Normal University, 1, 43-48.

[11] Ballı, S., & Karasoy, O. (2019). Development of content-based SMS classification application by using Word2Vec-based feature extraction. IET Software, 13(4), 295-304.

[12] Stephan Gouws. (2015). Word2vec: How can hierarchical soft-max training method of CBOW guarantee its self-consistence? URL: https://www.quora.com/Word2vec-How-can-hierarchical-soft-max-training-method-of-CBOW-guarantee-its-self-consistence.