





# Using PMI to Rank and Filter Edges in Graphs of Words

Marcela Ribeiro de Oliveira<sup>(✉)</sup> and Eduardo J. Spinosa<sup></sup>

Federal University of Paraná, Curitiba, Brazil  
{mroliveira, spinosa}@inf.ufpr.br

**Abstract.** Text classification is a classic problem in Natural Language Processing. An essential task in text classification is the construction of the representation, which must provide relevant information to the classifier. One of the most effective representation models uses graphs to represent documents. This paper presents an approach that uses this representation model but with weighted graphs. We propose to use a popular word association measure, the Pointwise Mutual Information (PMI), to calculate the weights of graph edges. These weights then serve as a guide to identify and remove edges between words with low association levels. Then, using node2vec, we extract the features of each graph and use a text convolutional neural network for classification. We conducted experiments in order to compare different kinds of graph modeling in terms of classification score and the proportion of edges that were removed. The results obtained indicate that this approach makes it possible to reduce the number of edges in the graphs maintaining classification performance.

**Keywords:** Graph of words · Word association · Edge filtering

## 1 Introduction

Text classification is a widely explored problem in the Natural Language Processing (NLP) field. It can be applied in several tasks like document organization, spam filtering and news filtering. In general, text classification has the following steps: data processing, text processing, feature extraction, training and evaluation of the model. In particular, feature extraction transforms the data into a representation that the algorithms can process. Among the existing forms of text representation, such as bag-of-words and n-grams, one of the most effective is based on graphs, which is the focus of this work.

Graphs can capture important information in text, such as term co-occurrence and term relationships, which are not considered by the bag-of-words model [16]. One possible transformation of a text to a graph, known as graph of words, considers words as nodes and their co-occurrence inside a sliding window of fixed size as edges. A graph representation makes it possible to use a node embedding algorithm, that maps each graph node to a feature vector and uses these feature vectors for classification.

In this paper we present an approach to remove edges from text graphs without causing loss in text classification performance. For this, we propose the use of Pointwise

Mutual Information (PMI), an existing co-occurrence measure based on word association, to calculate graphs' weights. Then, ranking the edges by their weight we can remove the edges containing the lower weights.

By identifying and removing redundant graph edges we aim to generate a representation that, despite containing fewer edges than graphs generated by co-occurrence alone, does not lead to a loss in the classification score. In this way, it is possible to represent the same document by a graph using fewer edges.

Our approach also allows to define the percentage of edges to be removed, which we call cut percentage  $p$ . Thus, through experiments with several values for the cutting percentage  $p$ , it is possible to define how many graph edges can be removed without causing significant loss in the representation.

We performed experiments in four text classification datasets that encompass different text classification tasks. The results obtained indicate that this approach makes it possible to reduce the number of edges in the graphs maintaining classification performance.

The rest of this paper is organized as follows. Section 2 gives an overview of the related work using graphs to represent texts. Section 3 provides details of our proposed approach. Section 4 presents the experiments methodology, including implementation details, the datasets employed and the evaluation process. Section 5 presents the results and their discussion and Sect. 6 draws conclusions and describes future work.

## 2 Related Work

Graphs have been used to represent text in several tasks, including emotion recognition [6], sentiment classification [8, 19], text generation [12] and question answering [5].

In the text classification problem specifically, graphs are used to represent documents even when deep neural network models are not used, which is the case of [15] and [17]. In Rousseau et al. [15], each text is represented using the graph of words model, so they treated the problem as a graph classification problem. In this approach, frequent subgraphs were extracted and used as features for a linear SVM classifier. Skiannis et al. [17] also proposed graphs that represent collections and labels. A collection graph is the union of all graphs that belong to a collection, and a label graph represents a class in which all the words of documents belonging to the respective class are the nodes and their co-occurrence are the edges.

Recently several works have been conducted using deep learning and graphs to text classification [9, 11, 14, 18]. The one that is closest to ours was proposed by Bijari et al. [1]. As the before-mentioned works, each sentence is represented by a graph. In this research, node2vec is applied to the sentence graphs to obtain the feature representation in an unsupervised manner. Then, a slight variant of the ConvNet architecture of Kim [10] and Collobert et al. [3] is used for sentiment classification.

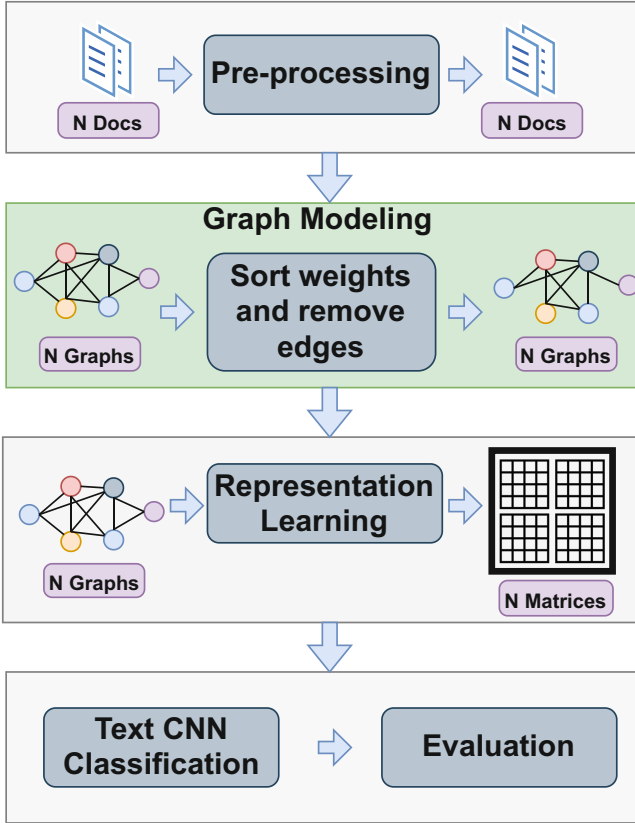
These works motivated ours to investigate the graph construction and evaluate the impact of removing edges in the graph representation learning process in order to obtain a more concise representation.

## 3 Proposed Approach

The proposed approach considers a classification problem where each document is represented by a graph. In the experiments, we compare unweighted graphs with all

co-occurrence edges with weighted graphs where a predefined percentage of edges were removed. We used node2vec to extract the representation of the graphs and then a convolutional neural network to perform the classification.

Figure 1 presents an overview of our approach composed of the following steps: text pre-processing, graph modeling, representation learning and classification. Next, we describe each of these components.



**Fig. 1.** Proposed approach overview. First, we pre-process the documents and model each of them as a graph. Then we sort the edges by weight and remove a certain percentage of them. Finally, we generate the embedding matrices and classify the documents with the Text CNN.

### 3.1 Text Pre-processing

In this step, transformations are applied in the documents, so they can be adequately represented and processed by an algorithm. In our approach, we have adopted the following text processing techniques: lowercase conversion, tokenization, non-alphabetic characters and stop words removal and stemming.

### 3.2 Graph Modeling

The graphs in our approach follow the graph of words model [15]: each document is represented by a graph, the nodes are the unique terms of the document, and the edges represent the co-occurrences of the terms within a sliding window of fixed size. This graph is also known as the co-occurrence graph.

In our work, we generated two representation models to be compared, one with unweighted and another with weighted graphs; the first one is a co-occurrence graph only, the other one has edge weights calculated by a co-occurrence word association measure. This kind of measure estimates the association between two words by comparing the word pair’s corpus-level bigram frequency to some function of the unigram frequencies of the individual words [4].

A popular word association measure is the Pointwise Mutual Information (PMI) [2], calculated by Eq. 1:

$$\mathbf{PMI} = \log \frac{f(x, y)}{f(x) * f(y) / W} \quad (1)$$

$W$  represents the number tokens in the corpus;  $f(x)$ ,  $f(y)$  are the unigram frequencies of  $x$  and  $y$  words respectively in the corpus; and  $f(x, y)$  is the  $(x, y)$  word pair frequency with amplitude restriction (frequency of co-occurrence of the word pair within a fixed size window).

Based on PMI, we generate two weighted graphs: **Local PMI** and **Global PMI**, that differ in how the weight is computed. Local PMI considers a single document in the computation, so the same word pair can have different values depending on the document that generated the graph. And Global PMI considers the set of all documents in the computation so that the same word pair will have the same values for all graphs.

After the computation of the weights, for each graph we sort the edges by their weight and remove a certain percentage  $p$  of the edges starting from the ones with lower values, where  $p$  is a parameter of the algorithm. If that process results in the disconnection of a word (vertex) from the graph, the vertex is removed and not considered by the representation learning algorithm. This implies that the graph that represents the text may have fewer words than the original text.

We expect that this method for edge removal guided by a word association measure leads us to find a more concise representation maintaining quality. In other words, if there is no loss in the representation quality using the proposed method to remove edges, it means that we can represent a text by a graph using less edges than the ones captured by the co-occurrence window alone. And also that the measure that we propose to apply is useful in ranking the edges by relevance to the representation and finding the unnecessary ones.

### 3.3 Graph Representation Learning

We used a node embedding algorithm to extract the graph’s representation. The idea of this type of algorithm is to automatically learn to encode the graph structure into low-dimensional embeddings. So, every graph node is represented by an embedding.

We chose node2vec [7], a semi-supervised graph representation learning algorithm that works through random walks starting from each vertex of the graph. Two node2vec characteristics are: a flexible notion of the neighborhood of a vertex and a random biased walk, since it is guided by parameters, in which several neighborhoods of the vertex are explored [7].

Since node2vec gives one embedding per graph node, an embedding matrix  $E_{n \times d}$  represents an entire graph, where  $n$  is the number of nodes, and  $d$  is the embedding dimension. Moreover, since each document is represented by a graph, we have one embedding matrix for each document.

### 3.4 Text CNN for Classification

For the classification step, we use a convolutional neural network (CNN), based on the Text CNN proposed by Kim [10], which receives an embedding matrix where each line is a word embedding from the document. The main difference compared to Kim’s approach is that, instead of having an embedding matrix with word embeddings, our approach will have node embeddings.

To ensure that all embedding matrices have the same size before submitting them as an input to the CNN, we calculated a threshold value to ensure that the matrix would fit 75% of the documents in the dataset. Texts larger than this value were truncated, and the smaller ones were zero-padded.

## 4 Experiment Setup

This section provides details about how we conducted the experiments, including implementation, datasets and evaluation methodology.

The experiments were performed in a server running Linux Mint version 19.5 with 16 cores of Intel(R) Xeon(R) E5620 processors and 70 GB of RAM.

### 4.1 Implementation Details

Since in the graph representation learning step we used node2vec, we had to set the values for some parameters. Table 1 presents these parameters, a brief description of each one of them and the value used for all the experiments.

**Table 1.** node2vec parameters, their descriptions and the values used in the experiments

Parameter	Description	Value
Dimensions	Embeddings dimension	100
walk_length	Number of nodes in each walk	2
num_walks	Number of walks per graph node	10
p	Return parameter	2
q	In-out parameter	0.7
Workers	Number of workers for parallel execution	4

For the classification step, the Text CNN had six convolutional layers, three with kernel size 2 and filter sizes 32, 64 and 128, and three with kernel size 3 and filter sizes 32, 64 and 128. The feature maps generated by all six convolutional blocks were globally max-pooled and fed into a fully connected layer with 256 neurons. We also included a dropout layer after each convolutional layer and one before the fully connected layer to avoid overfitting. The fully connected layer output was fed into a softmax layer for classification. We trained the Text CNN for 20 epochs.

## 4.2 Datasets

We evaluate our approach across two different text classification tasks: sentiment analysis and topic classification. For this, we chose the following datasets, which are widely used in text classification literature:

- **Polarity v2.0**: a sentiment analysis dataset composed by 2000 movie-reviews, where 1000 are positive and 1000 are negative.
- **WebKB**: a topic classification dataset containing web pages collected from computer science departments. The original dataset contains 7 classes, but we used a subset with 4 classes (student, faculty, course, project), composed of 7287 documents.
- **R8**: a topic classification dataset containing news documents labeled into 8 categories. This is a subset of the Reuters-21578 dataset [13].
- **20-Newsgroups (20NG)**: a topic classification dataset which contains approximately 20000 newsgroup posts, partitioned (nearly) evenly across 20 different topics. We removed the duplicated news documents, resulting in 18846 documents.

Table 2 summarizes the characteristics of each dataset.

**Table 2.** Dataset characteristics

Dataset	Documents	Classes
Polarity	2000	2
WebKB	7287	4
R8	7674	8
20 Newsgroups	18846	20

## 4.3 Evaluation

To preserve the percentage of samples of each class, we used stratified cross-validation. Also, to ensure that all the folds had the same samples in all experiments, we fixed the seed in fold generation. We split each dataset in ten folds. For each fold, 90% of the samples were used to train, and 10% of the samples were used to test.

To evaluate the classification performance we used F1-score. As we are using 10-fold cross-validation, we have ten F1-scores, one for each fold, so we report the mean of these ten F1-scores in the results to give a single estimation.

To evaluate if there is a significant difference between the classification score in the graphs constructed using our approach to remove edges compared to the unweighted graphs without edge removal, we performed a statistical test. We chose the Wilcoxon signed-rank test, a non-parametric test, where the null hypothesis is that two related paired samples come from the same distribution. We rejected the null hypothesis when the  $p$ -value was less than 0.05.

## 5 Results and Discussion

We compare the following representations: Unweighted graphs ( $\overline{W}$ ) against Local PMI weighted graphs, and Unweighted graphs ( $\overline{W}$ ) against Global PMI weighted graphs. For each representation, we generate graphs with window sizes of 4, 12 and 20. And for each combination of graph type and window size we evaluate three values for the cut percentage  $p$ : 5%, 10% and 20%.

Table 3 presents the mean F1 classification scores over 10 folds (cross-validation) for each dataset and parameter setting. Then, we apply the Wilcoxon test to verify if there is statistical difference between the baseline representation ( $\overline{W}$ ) and the proposed approach with edge removal (either using Local or Global PMI). Only the classification scores in bold are statistically different: either better (marked with “+”) or worse (marked with “-”) than the baseline. Our main interest is to maintain the classification score even with a smaller number of edges, indicating that these removed edges were not necessary to maintain the quality of the representation.

**Table 3.** 10-fold mean F1-score for each dataset, window size and cut percentage using unweighted graphs ( $\overline{W}$ ), weighted graphs with Local PMI and weighted graphs with Global PMI.

		Polarity			WebKB			R8			20 NG			
		Window size			Window size			Window size			Window size			
	Graph	Cut $p$	4	12	20	4	12	20	4	12	20	4	12	20
Baseline	$\overline{W}$	0	78.98	78.28	76.42	89.88	89.58	90.15	81.42	82.82	81.09	83.19	82.73	83.61
Proposed approach	Local PMI	5	77.73	78.49	78.71	90.48	89.10	89.87	81.47	81.83	<b>85.40+</b>	83.05	82.64	<b>82.63-</b>
		10	78.83	78.76	<b>80.04+</b>	89.80	89.48	89.90	80.82	80.27	82.44	83.07	82.68	<b>82.73-</b>
		20	79.93	79.01	77.84	89.14	89.34	89.31	79.29	81.74	81.83	<b>81.24-</b>	<b>82.01-</b>	<b>82.56-</b>
	Global PMI	5	78.63	76.93	78.63	89.68	88.92	88.63	79.65	80.10	<b>85.22+</b>	83.06	82.51	83.17
		10	79.21	77.22	76.68	89.67	88.56	<b>88.11-</b>	80.57	80.39	82.57	83.46	<b>83.66+</b>	83.00
		20	79.19	76.09	79.48	89.71	88.69	89.37	80.45	81.21	82.02	83.51	83.07	83.09

In general, we can observe that the results were stable for most of the datasets, except for the 20 Newsgroups, where there were many statistically worse results using Local PMI specifically. The WebKB dataset also obtained one statistically worse result, but this time using Global PMI to calculate edge weights. Polarity and R8 datasets were those who have obtained more stability, presenting only positive statistically significant variations.

Observing the window size parameter, we can conclude that larger windows do not improve the classification score. And smaller windows sizes, like 4 and 12, present more stability in terms of the classification score.

The weight measure Global PMI performs better than the Local PMI, and it is possible to observe that when we use Global PMI with window size 4, even removing up to 20% of the edges, there is no statistically significant loss in the F1-score.

In this way, we can conclude that it is possible to calculate the edges' weights from the co-occurrence graph using PMI and sort these weights in order to remove a percentage of them from the graph, without causing a loss in the representation quality. Also, the experiments pointed that generating graphs with window size 4, and Global PMI to calculate the edges' weights, it is possible to use graphs with 20% less edges and still obtain a statistically equivalent classification score.

## 6 Conclusion and Future Work

In this paper, we presented an approach to rank and filter edges from graphs of words. Based on a popular word association measure, the Pointwise Mutual Information (PMI), we propose two ways to calculate the edge weights: the Local and Global PMI measures. In this context, our approach first ranks graph edges by their weights and then removes the smaller ones from the graph until it reaches a fixed percentage  $p$ .

The results showed that both measures, Local PMI and Global PMI, help to improve the representation in the sense of identifying edges that can be removed from the graph without causing a loss in the classification score. Moreover, the Global PMI measure generated more stable results.

Future work includes: exploring other word association measures to calculate the graph-of-words' edge weights; evaluating more cut thresholds in order to get the best value to remove edges without causing a statistically significant loss of representation quality; and evaluating and comparing graph representation learning time and memory usage for the graphs constructed using our approach and the unweighted graphs.

**Acknowledgements.** We thank the financial support from the Brazilian National Council for Scientific and Technological Development (CNPq). We also would like to thank Alana AI for its financial support.

## References

1. Bijari, K., Zare, H., Kebriaei, E., Veisi, H.: Leveraging deep graph-based text representation for sentiment polarity applications. *Expert Syst. Appl.* **144**, 113090 (2020)
2. Church, K.W., Hanks, P.: Word association norms, mutual information, and lexicography. *Comput. Linguist.* **16**(1), 22–29 (1990)
3. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **12**(1), 2493–2537 (2011)
4. Damani, O.: Improving pointwise mutual information (PMI) by incorporating significant co-occurrence. In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, Sofia, Bulgaria, pp. 20–28. Association for Computational Linguistics, August 2013. <https://www.aclweb.org/anthology/W13-3503>



5. De Cao, N., Aziz, W., Titov, I.: Question answering by reasoning across documents with graph convolutional networks. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota, pp. 2306–2317. Association for Computational Linguistics, June 2019. <https://doi.org/10.18653/v1/N19-1240>, <https://www.aclweb.org/anthology/N19-1240>
6. Ghosal, D., Majumder, N., Poria, S., Chhaya, N., Gelbukh, A.: DialogueGCN: a graph convolutional neural network for emotion recognition in conversation. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, pp. 154–164. Association for Computational Linguistics, November 2019. <https://doi.org/10.18653/v1/D19-1015>, <https://www.aclweb.org/anthology/D19-1015>
7. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM (2016)
8. Huang, B., Carley, K.: Syntax-aware aspect level sentiment classification with graph attention networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, pp. 5469–5477. Association for Computational Linguistics, November 2019. <https://doi.org/10.18653/v1/D19-1549>, <https://www.aclweb.org/anthology/D19-1549>
9. Huang, L., Ma, D., Li, S., Zhang, X., Wang, H.: Text level graph neural network for text classification. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, pp. 3444–3450. Association for Computational Linguistics, November 2019. <https://doi.org/10.18653/v1/D19-1345>, <https://www.aclweb.org/anthology/D19-1345>
10. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, pp. 1746–1751. Association for Computational Linguistics, October 2014. <https://doi.org/10.3115/v1/D14-1181>, <https://www.aclweb.org/anthology/D14-1181>
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (ICLR) (2017)
12. Koncel-Kedziorski, R., Bekal, D., Luan, Y., Lapata, M., Hajishirzi, H.: Text generation from knowledge graphs with graph transformers. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota, pp. 2284–2293. Association for Computational Linguistics, June 2019. <https://doi.org/10.18653/v1/N19-1238>, <https://www.aclweb.org/anthology/N19-1238>
13. Lewis, D.D.: An evaluation of phrasal and clustered representations on a text categorization task. In: Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 37–50 (1992)
14. Linmei, H., Yang, T., Shi, C., Ji, H., Li, X.: Heterogeneous graph attention networks for semi-supervised short text classification. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, pp. 4821–4830. Association for Computational Linguistics, November 2019. <https://doi.org/10.18653/v1/D19-1488>, <https://www.aclweb.org/anthology/D19-1488>

15. Rousseau, F., Kiagias, E., Vazirgiannis, M.: Text categorization as a graph classification problem. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Beijing, China, pp. 1702–1712. Association for Computational Linguistics, July 2015. <https://doi.org/10.3115/v1/P15-1164>, <https://www.aclweb.org/anthology/P15-1164>
16. Shanavas, N., Wang, H., Lin, Z., Hawe, G.: Structure-based supervised term weighting and regularization for text classification. In: Métails, E., Meziane, F., Vadera, S., Sugumaran, V., Saraee, M. (eds.) NLDB 2019. LNCS, vol. 11608, pp. 105–117. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-23281-8\\_9](https://doi.org/10.1007/978-3-030-23281-8_9)
17. Skianis, K., Malliaros, F., Vazirgiannis, M.: Fusing document, collection and label graph-based representations with word embeddings for text classification. In: Proceedings of the Twelfth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-12), New Orleans, Louisiana, USA, pp. 49–58. Association for Computational Linguistics, June 2018. <https://doi.org/10.18653/v1/W18-1707>, <https://www.aclweb.org/anthology/W18-1707>
18. Yao, L., Mao, C., Luo, Y.: Graph convolutional networks for text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 7370–7377 (2019)
19. Zhang, C., Li, Q., Song, D.: Aspect-based sentiment classification with aspect-specific graph convolutional networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, pp. 4560–4570. Association for Computational Linguistics, November 2019. <https://doi.org/10.18653/v1/D19-1464>, <https://www.aclweb.org/anthology/D19-1464>