Text classification problems via BERT embedding method and graph convolutional neural network

Loc TRAN
CHArt Laboratory
EPHE - Paris-Saclay University
Paris, France
tran0398@umn.edu

Lam PHAM
University of Information Technology, HCMC, Vietnam
Vietnam National University, Ho Chi Minh city, Vietnam
16520645@gm.uit.edu.vn

Tuan TRAN
CHArt Laboratory
EPHE - Paris-Saclay University
Paris, France
anh-tuan.tran@etu.ephe.psl.eu

An MAI*
Department of Computer Science & Engineering
International University, Ho Chi Minh city, Vietnam
Vietnam National University, Ho Chi Minh city, Vietnam
mhban@hcmiu.edu.vn

***Abstract*—This paper presents a clever technique of combining the BERT embedding method and the graph convolutional neural network. This combination is then employed to solve the text classification problem. Initially, we apply the BERT embedding method to the whole corpus in order to transform all the texts into numerical vectors. Then, the graph convolutional neural network will be applied to these numerical vectors to classify these texts into their appropriate classes. Especially, in our approach, we need only few labeled texts for the model training. For the illustration, in this paper, we use the BBC news and the IMDB movie reviews datasets to perform our experiments, showing that the performance of the graph convolutional neural network model is better than the performances of the combination of the BERT embedding method with other classical machine learning models.**

***Index Terms*—text classification, graph convolutional neural network, BERT, classical machine learning model**

I. Introduction and related works

Text classification is the very important problem in machine learning and deep learning research areas. Its applications are huge such as emotion/sentiment classification [1], intent classification [2], and genre text classification [3], etc. The outcome of this text classification problem is very valuable. Let's discuss deeply about the advantages and the motivations of one specific text classification problem which is the genre text classification problem. After classifying one company's products (such as video or music clips on YouTube and Facebook) into their appropriate classes/types/labels based on the descriptions of the products, we can achieve a lot of goals such as:

- Recommendation system/Analyze the match fit among users/products: This will increase the engagement of users/fans to the activities of company such as view/buy/listen to ...,
- Resource allocation/Optimization/Decision Making: We do know in advance which types of products that users/fans want to view/buy (and don't want to view/buy). We can put more times/human resources to produce those types of products that the users/fans want to view/buy ...,
- Organize all the products neatly and meaningfully. This will help the "search for products" button/function ...,
- We know the trends/favors of users/fans (in our network/our community). These trends/favors are very important and are very valuable.

Thus, we can easily see that text classification problem is the very valuable problem for any business. In order to solve this text classification problem, we can employ three types of approaches: the rules-based approach is based on a huge collection of text-rules on words, phrases, sentences, etc., or in general, a brach of computational linguistic. The second approach relies on the classical machine learning techniques and he last one is the modern deep learning approach when we simply do not care about the text-rules but a huge of labeled contents, basically.

One of the wisest technique in modern approach is representing the text content in the finite high dimensional spaces before input them to the learning models. For more detail, initially, we need to transform the texts (the descriptions of the data samples or the comments of the users or fans) to numerical vectors. By using the classical embedding approach, we can utilize the "tf-idf" method [4] or count-vectorize method [5] or the combination of these two methods to transform the texts to numerical vectors. By using the modern embedding approach, we can utilize the BERT embedding method [6], the ELMO embedding method [7], or the FLAIR embedding method [8] to transform the texts to numerical vectors.

Finally, after obtaining the numerical vectors from texts, we can employ the classical machine learning models such as the logistic regression model [9], the random forest model [10], and the support vector machine model [11] to classify the texts into their appropriate classes/labels. Or, we can employ the

*Corresponding author.

modern deep learning models such as the deep convolutional neural network model [12] to classify the texts into their appropriate classes/labels.

However, there is one weakness associated with the modern deep learning models. In this case, for example, although the deep convolutional neural network model can achieve significant performance, this model normally ignores the relationships (i.e. the connections) among the samples (i.e. texts) which are very popular in text data. Recently, many research approaches focus on the use of graph structure to model the relations in data to improve the performance of learning problem [13]–[15].

In order to overcome this weakness, we propose to employ the graph convolutional neural network model [16]–[19] to solve this text classification problem. Obviously, this graph convolutional neural network model utilizes both types of information:

- The numerical vectors (for example, the BERT embedding vectors) of texts,
- The graph data structure representing the relationships among the texts.

In this paper, our contributions are three folds:

- Utilize the graph constructed from the BERT embedding vectors of the texts to solve the text classification problem,
- Employ the graph convolutional neural network model to solve the text classification problem (utilize both the BERT embedding vectors and the graph data structure representing relationships among the texts),
- Compare the performance of the graph convolutional neural network model with the performance of the combinations of the BERT embedding method with classical machine learning models.

We will organize the paper as follow: In section I, we introduce the problem and review some notable related works. Then section II will define the problem and will show the way how to construct the graph from the BERT embedding vectors of texts and will present the graph convolutional neural network. Next, section III will describe the two datasets that will be used in this paper which are the BBC news dataset and the IMDB movie reviews dataset and will show how to construct the graphs from the two datasets in detail and will compare the performance the graph convolutional neural network model and the performance of the combinations of the BERT embedding method and the classical machine learning models such as linear regression model testing on these two datasets. Finally, section IV will conclude this paper and the future directions of researches will be discussed.

II. Problem Formulation and Graph Convolutional Neural Network

A. Problem Formulation

Given the set of the texts $\{x_1, x_2, \ldots, x_l, x_{l+1}, \ldots, x_{l+u}\}$ where $n = l + u$ is the total number of texts.

Note that $\{x_1, x_2, \ldots, x_l\}$ is the set of labeled texts and $\{x_{l+1}, \ldots, x_{l+u}\}$ is the set of un-labeled texts, in which, $x_i \in R^{1*L_1}, 1 \leq i \leq n$. L_1 is the number of input features (i.e., the dimensions of the BERT embedding vectors). In the other words, x_i is the BERT embedding vectors of the text i.

Let C be denoted for the total number of classes/labels, and let $Y \in R^{n*C}$ be the initial label matrix. Y can be defined as follows

$$Y_{i,j} = \begin{cases} 1 & \text{if text } i \text{ belongs to class } j, 1 \leq i \leq l \\ 0 & \text{if text i does not belongs to class } j, 1 \leq i \leq l \\ 0 & l+1 \leq i \leq n \end{cases} \tag{1}$$

Our objective is to predict the labels of the un-labeled texts $x_{l+1}, \ldots, x_{l+u}$.

B. Construct the graph from the BERT embedding vectors

After transforming all the texts in the dataset to numerical vectors by using the BERT embedding method, we can construct the similarity graph for these text vectors by using the 5-nearest neighbor graph. In the other words, text i is connected with text j by an edge (no direction: un-directed graph) if text i is among the 5 nearest neighbors of text j **or** text j is among the 5 nearest neighbor of text i.

Finally, we obtain the adjacency matrix A representing for the similarity graph.

$$A_{i,j} = \begin{cases} 1 & \text{if text } i \text{ is connected to text } j \\ 0 & \text{if text } i \text{ is not connected to text } j \end{cases} \tag{2}$$

C. The graph convolutional neural network model

Currently, we have the set of BERT embedding vectors $\{x_1, x_2, \ldots, x_n\}$ and the adjacency matrix A representing the relationships among the texts in the dataset.

Note that $x_i \in R^{1*L1}, 1 \leq i \leq n$ and $A \in R^{n*n}$

Let $\hat{A} = A + I$, where I is the identity matrix. By using this adjustment, we can always guarantee the invertibility of the matrix $\hat{A}$.

Let $\hat{D}$ be the diagonal degree matrix of $\hat{A}$. In the other words, $\hat{D}_{ii} = \sum_j \hat{A}_{i,j}$.

The final output Z of the graph convolutional neural network can be learned as follows

$$Z = \mathit{softmax}\ (\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}\mathit{ReLU}(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}X\theta_1)\theta_2). \tag{3}$$

Note that X is known as the input feature matrix and $X \in R^{n*L_1}$. In the learning process, $\theta_1 \in R^{L_1*L_2}$ and $\theta_2 \in R^{L_2*C}$ are two-parameter matrices that are needed to be learned during the training process.

In this work, we apply *ReLU* (also called Rectified Linear Unit) activation function. It is defined as in the following equation:

$$\mathit{ReLU}(x) = max(0, x). \tag{4}$$

Normally, in the end of the training process, the *softmax* operation will be considered to output the class probabilities, taking an input vector z of K real numbers. In the other words, $z \in R^K$ and $z = [z_1, z_2, \ldots, z_K]$.

Herein, the *softmax* operation can be defined as follow

$$\mathit{softmax}(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}. \tag{5}$$

For semi-supervised multiclass classification [20]–[29], we can then evaluate the cross-entropy error over all labeled texts:

$$L = -\sum_{k=1}^{l}\sum_{c=1}^{C} Y_{kc} \ln Z_{kc}. \tag{6}$$

The two parameter matrices $\theta_1 \in R^{L_1 * L_2}$ and $\theta_2 \in R^{L_2 * C}$ are trained by using the gradient descent method, and updated sequentially over the training process.

In our work, this semi-supervised text classification problem is not like the normal supervised classification problem. In the other words, we just need a few texts with labels (for examples, we just need 10 to 100 texts with labels) [20]–[29]. This fact also means that we do not need thousands of labeled samples to construct the semi-supervised learning models. Then, we can apply the graph convolutional neural network to classify a couple of thousands of unlabeled texts to their appropriate classes/labels. This is the very strong argument of the semi-supervised learning framework. Last but not least, the performance of the graph convolutional neural network is expected to be better than the performance of the combinations of the BERT embedding method and the classical machine learning models. This statement will be illustrated in the Experiments and Results section.

III. Experiments and Results

A. Dataset descriptions

To show a fair comparison, we will use two different datasets which are the BBC news dataset and the IMDB movie reviews dataset to test our approaches.

The BBC news dataset contains 2126 short texts. These 2126 short texts belong to 5 classes which are sport, business, politics, tech, and entertainment. After pre-processing these short texts, we apply the BERT embedding method to these short texts in order to transform these short texts to numerical vectors. In the other words, we will get 2126 numerical vectors R^{1024}. Finally, we need to construct the similarity graph from these 2126 numerical vectors before applying classification models to these 2126 BERT embedding vectors (and the similarity graph).

The IMDB movie reviews dataset contains 5000 short texts. These 5000 short texts belong to 2 classes which are 1 (positive/good review) and 0 (negative/bad review). After pre-processing these short texts, we apply the BERT embedding method to these short texts in order to transform these short texts to numerical vectors. In the other words, we will get 5000 numerical vectors R^{1024}. Finally, we need to construct the similarity graph from these 5000 numerical vectors before applying classification models to these 5000 BERT embedding vectors (and the similarity graph).

There are three ways to construct the similarity graph from the BERT embedding vectors such as:

- The ϵ-neighborhood graph: Connect all the texts whose pairwise distances are smaller than ϵ,
- k-nearest neighbor graph: Text i is connected with text j by an edge (no direction: un-directed graph) if text i is among the k nearest neighbors of text j **or** text j is among the k nearest neighbor of text i,
- The fully connected graph: All texts are connected.

In this paper, the k-nearest neighbor graph is employed to construct the similarity graph from the BERT embedding vectors of the BBC news dataset and the IMDB movie reviews dataset. Note that, without loss of generality, $k = 5$ is set in our experiments.

B. Experimental results

In this section, we will try to compare the accuracy performance of the graph convolutional neural network model with the accuracy performance of the combinations of the BERT embedding method with the classical machine learning models which are the logistic regression model, the random forest model, and the support vector machine model. We test our models (Python code) on Google Colab with NVIDIA Tesla K80 GPU and 12 GB RAM. Moreover, to demonstrate learning ability of our semi-supervised setting, we execute the experiments with the number of labeled texts only in the range from 50 to 110. For evaluation the learning model, we consider the accuracy performance Q, defined as follows:

$$Q = \frac{TP + TN}{TP + FP + TN + FN} \tag{7}$$

with True Positive (TP), True Negative (TN), False Positive (FB), False Negative (FN) are defined in the following table I.

Table I: Definitions of TP, TN, FP, FN

		Predicted Label	
		Positive	Negative
Known label	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FN)	True Negative (TN)

For the BBC news dataset, the accuracy performances of the graph convolutional neural network model, the combinations of the BERT embedding method and the logistic regression model, the random forest model, and the support vector machine model are given in the following table II. It can be easily observed that our proposed combination is remarkably better than the other combination, especially when the number of labeled texts are getting smaller.

For the IMDB movie reviews dataset, the accuracy performances of the graph convolutional neural network model, the combinations of the BERT embedding method and the logistic regression model, the random forest model, and the support vector machine model are given in the following table III. As we can see, the combination of BERT and graph convolutional

Table II: BBC news dataset: Comparison of graph convolutional neural network model with the combination of BERT embedding method with classical machine learning models. Accuracies (%) is reported.

Number of labeled texts	10	20	30	40	50
Graph Convolutional Neural Network	**76.42**	**87.70**	**91.79**	**90.17**	**90.22**
Logistic Regression	55.81	79.87	86.59	87.78	88.01
Random Forest	35.30	63.20	79.91	81.59	82.23
Support Vector Machine	62.24	80.86	86.64	87.20	88.54

Table III: IMDB movie reviews dataset: Comparison of graph convolutional neural network model with the combination of BERT embedding method with classical machine learning models. Accuracies (%) is reported.

Number of labeled texts	70	80	90	100	110
Graph Convolutional Neural Network	**73.79**	**73.49**	**73.40**	**73.95**	**74.51**
Logistic Regression	71.05	71.52	71.69	72.47	72.17
Random Forest	56.77	63.77	64.42	63.49	61.39
Support Vector Machine	68.58	68.23	68.82	71.61	69.84

neural network is always better than the the other approaches. Nevertheless, its performance is not really impressive due to the limitation of our training resources.

C. *Discussion*

From the above tables II and III, we easily recognize that, although with a limited training resources, the combination of BERT and graph convolutional neural network model is slightly or even significantly better the combinations of BERT embedding method with the logistic regression model, the random forest model, and the support vector machine model. This emphasizes that such semisupervised learning model like graph convolutional neural network can learn better the relations between the text contents. This semi-supervised learning model, in addition, works perfectly even when the labeled text is scarce. However, when the number of labeled text in the training set is extremely large (i.e. a couple of millions labeled texts) and the text contents are homogeneous and simple, the performances of the classical supervised learning models such as logistic regression model might be sometimes better than the performance of the semi-supervised learning model, but this often infeasible in practice.

IV. Conclusion

In this paper, our contributions are clearly presented via three folds:

- A similarity graph from using the BERT embedding is constructed,
- then the graph convolutional neural network is performed to solve the text classification problem,
- a clear comparison of our proposed combination and the combination of BERT with the classical machine learning models is presented on two different datasets.

Even though the research task constructing the similarity graph from the BERT embedding vectors looks easy, it's the very hard task. In this premise work, we only consider building the similarity graph taking into account thousands of vectors. In the real life applications (or even more practical scenarios), we need to build the similarity graph from millions up to billions of vectors (i.e. representing for millions or billions of texts). In the implementation perspective, the way how we perform the task of building the graph is really naïve and sequential. In the future, we need to figure out how to achieve this task in the smarter way. For example, organize the code in parallel and/or employ some specific data structure such as k-d tree. This approach is not only important for research development but also practical for business application.

Moreover, to the best of our knowledge, there are various types of graph convolutional neural networks have been developed. In the next research works, we will employ the graph convolutional neural network with attention [17] and the hypergraph convolutional neural network [18] to solve the text classification problem.

References

[1] L. G. Singh, A. Mitra, and S. R. Singh, "Sentiment analysis of tweets using heterogeneous multi-layer network representation and embedding," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 8932–8946.

[2] S. Larson, A. Mahendran, J. J. Peper, C. Clarke, A. Lee, P. Hill, J. K. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang *et al.*, "An evaluation dataset for intent classification and out-of-scope prediction," *arXiv preprint arXiv:1909.02027*, 2019.

[3] M. Ikonomakis, S. Kotsiantis, and V. Tampakas, "Text classification using machine learning techniques." *WSEAS transactions on computers*, vol. 4, no. 8, pp. 966–974, 2005.

[4] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, vol. 242, no. 1. Citeseer, 2003, pp. 29–48.

[5] S. M. Sekhar, G. Siddesh, M. Raj, and S. S. Manvi, "Protein class prediction based on count vectorizer and long short term memory," *International Journal of Information Technology*, vol. 13, no. 1, pp. 341–348, 2021.

[6] Z. Gao, A. Feng, X. Song, and X. Wu, "Target-dependent sentiment classification with bert," *IEEE Access*, vol. 7, pp. 154 290–154 299, 2019.

[7] N. Reimers and I. Gurevych, "Alternative weighting schemes for elmo embeddings," *arXiv preprint arXiv:1904.02954*, 2019.

[8] A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf, "Flair: An easy-to-use framework for state-of-the-art nlp," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, 2019, pp. 54–59.

[9] K. Shah, H. Patel, D. Sanghvi, and M. Shah, "A comparative analysis of logistic regression, random forest and knn models for the text classification," *Augmented Human Research*, vol. 5, no. 1, pp. 1–16, 2020.

[10] T. Pranckevičius and V. Marcinkevičius, "Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification," *Baltic Journal of Modern Computing*, vol. 5, no. 2, p. 221, 2017.

[11] Z.-Q. Wang, X. Sun, D.-X. Zhang, and X. Li, "An optimal svm-based text classification algorithm," in *2006 International Conference on Machine Learning and Cybernetics*. IEEE, 2006, pp. 1378–1381.

[12] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," *arXiv preprint arXiv:1606.01781*, 2016.

[13] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, 2014.

[14] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.

[15] Q. Dang, A. Mai, M. Ngo, and T. Bui, “Rotat3d: A knowledge graph embedding using relational rotation in 3d vector space,” in *2020 12th International Conference on Knowledge and Systems Engineering (KSE)*. IEEE, 2020, pp. 201–206.

[16] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.

[17] Z. Guo, Y. Zhang, and W. Lu, “Attention guided graph convolutional networks for relation extraction,” *arXiv preprint arXiv:1906.07510*, 2019.

[18] N. T. V. Dang, L. Tran, and L. Tran, “Noise-robust classification with hypergraph neural network,” *arXiv preprint arXiv:2102.01934*, 2021.

[19] M. Ngo, A. Mai, and T. Bui, “On an improvement of graph convolutional network in semi-supervised learning,” in *2020 7th NAFOSTED Conference on Information and Computer Science (NICS)*. IEEE, 2020, pp. 114–117.

[20] L. Tran, “Application of three graph laplacian based semi-supervised learning methods to protein function prediction problem,” *arXiv preprint arXiv:1211.4289*, 2012.

[21] T. Loc, “Hypergraph and protein function prediction with gene expression data,” *arXiv preprint arXiv:1212.0388*, 2012.

[22] L. Tran, “The un-normalized graph p-laplacian based semi-supervised learning method and protein function prediction problem,” in *Knowledge and Systems Engineering*. Springer, 2014, pp. 23–35.

[23] H. Trang and L. H. Tran, “Graph based semi-supervised learning methods applied to speech recognition problem,” in *International Conference on Nature of Computation and Communication*. Springer, 2014, pp. 264–273.

[24] L. H. Tran and L. H. Tran, “Un-normalized graph p-laplacian semi-supervised learning method applied to cancer classification problem,” *Journal of Automation and Control Engineering Vol*, vol. 3, no. 1, 2015.

[25] L. Tran and L. Tran, “The un-normalized graph p-laplacian based semi-supervised learning method and speech recognition problem,” *International Journal of Advances in Soft Computing & Its Applications*, vol. 9, no. 1, 2017.

[26] L. H. Tran, T. Hoang, and B. H. N. Huynh, “Hypergraph based semi-supervised learning algorithms applied to speech recognition problem: a novel approach,” *arXiv preprint arXiv:1810.12743*, 2018.

[27] L. H. Tran and L. H. Tran, “Un-normalized hypergraph p-laplacian based semi-supervised learning methods,” *arXiv preprint arXiv:1811.02986*, 2018.

[28] L. Tran, A. Mai, T. Quan, and L. Tran, “Weighted un-normalized hypergraph laplacian eigenmaps for classification problems.” *International Journal of Advances in Soft Computing & Its Applications*, vol. 10, no. 3, 2018.

[29] L. H. Tran and L. H. Tran, “Directed hypergraph neural network,” *arXiv preprint arXiv:2008.03626*, 2020.