EEE 586: Survey for the Term Project Text Classification with Graph Neural Networks

Tuna Alikaşifoğlu
Dept. of Electrical and Electronics Engineering
Bilkent University
t.alikasifoglu@bilkent.edu.tr

Arda Can Aras
Dept. of Electrical and Electronics Engineering
Bilkent University
can.aras@bilkent.edu.tr

***Abstract*—In recent years, the amount of text based complex documents increased significantly, along with the importance of ability to classify texts as efficiently and accurately as possible. We had traditional algorithms to tackle the text classification problem, but with the progressive computational power, many machine learning, especially deep learning, based solutions surpassed the human capabilities in this area. On the other hand, we have the trending topic of graph neural networks. They are geometric extensions, i.e., extensions of traditional neural network architectures to the graph domain, and graph-structured data. In this survey, a brief overview of text classification problem and of graph neural networks are provided. Text classification overview covers the fundamental steps of a text classification task with the indication of graph neural network integration, where the graph neural network overview provides the reasons, challenges and how-tos of utilizing graph neural networks. Finally, we provide an overview of the related work that tackles the text classification problem with the graph neural networks, while comparing several approaches.**

***Index Terms*—Text Classification, Graph Neural Networks (GNNs), Graph Convolutional Networks (GCNs), Convolutional Graph Neural Networks (ConvGNNs).**

I. Introduction

In this survey, we have scrutinize the classic natural language processing task of text classification along with the trending approach of graph neural networks. We have divided the survey into two parts: *Previous Work* (Section II) and *Related Work* (Section III). In Section II, we provide an overview regarding the traditional text classification (Section II-A) and we provide an overview to graph neural networks (Section II-B). Both of these overviews are provided based on several survey papers for text classification [1]–[3] and for graph neural networks [4]–[7]. In Section II-A, we provide a basic definition for text classification, then we disintegrate text classification process to five steps to analyse the overall process, and we point out where the integration of graph neural networks can be made. In Section II-B, we provide an introduction to graph neural networks. Then, we analyse why we need graph neural networks, we provide several challenges on the route of obtaining graph based deep learning frameworks, and finally we provide bare-minimum steps in order to obtain a graph neural network based architecture, based on the aforementioned surveys, in this subsection. After this step, we are at a stage that we have provided background information both on the preliminary aspects of the overall term project, which are text classification and graph neural networks.

In Section III, we provide previous work directly related to our own topic of graph neural networks related to solution of the classical natural language processing task of text classification. Section III of the survey aims to investigate the related and previous work in the text classification with graph neural network field possibly with a historical order. All the papers mentioned in this section related with each other by references. Surprisingly, this structure can also be viewed as graph where each paper is node and and edges of the graphs as citation. There are also studies on this topic to predict structured citation trend [8]. Most of the papers in the Section III did not directly proposed to solve text classification task. However, nearly all of the papers presented Section III used text classification bench mark data sets to evaluate model performance.

II. Previous Work

A. Text Classification

In [2], *text classification* (text categorization) is defined as the "procedure of designating predefined labels for the text". The task is to assign labels or tags to the text based knowledge, i.e., textual units such as sentences, paragraphs and documents, where the labels are usually defined by humans, but can also be defined by the machine. This task is a fundamental part of Natural Language Processing (NLP), and it is significant to its applications such as sentiment analysis, question answering, text summarization, etc.. Text classification task can be partitioned into five phases as preprocessing, feature extraction, dimensionality reduction (optional), classifier selection and evaluation:

1) Preprocessing: Text preprocessing is a crucial prerequisite for a successful feature extraction, and summarized in [1] as follows. The input of the text classification frameworks consists of raw text data, which are in the form of a sequence of sentences. In this step, "cleaning" of the text datasets is performed to transform the data into a form that is suitable for feature extraction. The cleaning process is usually performed by tokenization, capitalization, slang and abbreviation handling, noise removal, spelling correction, stemming and lemmatization.

2) Feature Extraction: After preprocessing step, another crucial step, feature extraction step is necessary. In [1], this step explained as follows. Two common methods of text based feature extraction are weighted word and word embedding techniques. In the weighted word aspect, we have old techniques like bag-of-words and term frequency-inverse document frequency (TF-IDF). In the relatively recent aspect, we have the word embedding techniques like *word2vec*, *GloVe*, *FastText*, etc.

3) Dimensionality Reduction: The dimensionality reduction is an optional step of a text classification task, but based on the size of the dataset, it may be a must to have a computable result. In this aspect of the task, we try to reduce the dimensionality of the feature space while preserving the information of the original features space. Some possible dimensionality reduction techniques provided in [1] include (principal / independent) component analysis, linear discriminant analysis, non-negative matrix factorization, random projection, autoencoder and stochastic neighbor embedding.

4) Classifier Selection: As it is stated in [2], selecting the optimal classifier is the most important aspect of a text classification task. Currently we have both traditional and deep learning oriented classifiers. The traditional classifiers are based on the statistical analysis of the training data, and the deep learning classifiers are based on the neural networks. The main distinction between the traditional and deep learning based approaches can be described as follows: Good feature extraction methodology is crucial for the traditional classifiers. They obtain sample features by artificial methods and then make classifications based on these features. Hence, the performance of the traditional classifiers are mainly restricted by feature extraction. On the other hand, by making feature mapping via nonlinear transformations a part of the learning process, deep learning based classifier selection can integrate feature extraction aspect into the model fitting process.

Examples of both traditional and deep learning based approaches are provided in [1]–[3]: Some traditional classifiers are logistic regression, (kernel) support vector machine, Naive Bayes, k-nearest neighbors, decision tree, random forest, etc. On the other hand, the deep learning classifiers are usually based on the neural networks, such as deep feed forward neural networks, convolutional neural networks (CNNs), recurrent neural networks (RNNs), lately attention and transformer based models such as BERT [9] variations and fine tuning pre-trained language models [10], and finally what we will focus on, graph neural network (GNN) based models.

5) Evaluation: Evaluation is step that we understand how the our model performs under the given text classification task. As it is provided in [2], [3], there are several evaluation metrics that can be used to evaluate the performance of a supervised technique. The most common metrics are accuracy, F_β-score, micro/macro-averaging. Although we also have metrics like Matthews correlation coefficient and receiver operating characteristics (ROC). In order to evaluate the performance of our model, based on the provided techniques, we need to use labeled data, i.e., we need benchmark datasets like GLUE [11], TweetEval [12], among others.

B. Graph Neural Networks

In recent years, deep learning based solutions surpassed any approach on machine learning tasks such as image classification, video processing, speech recognition and natural language processing. In these tasks, the underlying data are usually represented in the Euclidean domain. However, each day the amount of non-Euclidean data increases, which are represented as represented by graphs to capture the underlying the complex relationships and interdependency between objects. Therefore, a need for deep learning methods that can manage graph structured data has emerged. In this context, the graph neural networks (GNNs) are born, and many of the deep learning approaches are converted to graph domain such as recurrent GNNs, convolutional GNNs (ConvGNNs) or graph convolutional networks (GCNs), graph autoencoders (GAE), graph reinforcement learning (GRL), graph adversarial methods and spatial-temporal GNNs, as summarized in [4]–[7].

1) Reasons to use GNNs: Hidden patterns residing under Euclidean data can be effectively obtained by traditional deep learning techniques. However, the increasing number of applications based on a non-Euclidean data structure enforces the necessity of graph based solutions. In this aspect, the following examples can be used to illustrate the benefits of having a graph based deep learning framework [5]:

- In e-commerce, highly accurate recommendation system can be achieved by using graph based deep learning techniques, since the interactions between users and products are a textbook example of graph structured data.
- For drug discovery in chemistry, we need to obtain the bioactivity of the molecules, where the molecules are modeled as graphs.
- Categorization of articles in a citation network, where the articles are linked to each other via "citationships", i.e., forming a graph structure.

2) Challenges to use GNNs: In order to have a graph domain deep learning framework, we need to overcome several challenges imposed by the complexity of the graph data. Due to the nature of graphs, when they are compared with Euclidean data, they can be irregular, they can have unordered nodes with different number of neighbors. Hence, many basic operations defined in Euclidean domain can be challenging to apply to the graph domain, e.g., convolution operation. In addition, one of the fundamental assumption we have in the existing machine learning algorithms is that the instances are independent of each other, although this assumption is not valid for graph data since each instance (node) is related to others by links of various types [5]. Some of the main challenges can be categorized as follows [6]:

a) Irregular structures of graphs: We have the *geometric deep learning problem* which is the inability to define basic operations like convolution and pooling in the graph domain, which are essential aspects of traditional CNNs.

b) Heterogeneity and diversity of graphs: We have many different properties that a single graph can have: graphs can be homogenous or heterogeneous, they can be weighted or unweighted, they can be directed or undirected, and they can be signed or unsigned. Furthermore, the tasks may consist of node-level problems such as node classification, link prediction or they can consist of graph-level problems such as graph classification or graph generation. Therefore, we need a spectrum of architectures to tackle all these problems one-by-one.

c) Large-scale graphs: As in the case of e-commerce and social networks, graph structured data can have a large number of nodes and edges. However, we still need appropriate algorithms to work on the graph structure without increasing the computational and time complexity too much.

d) Incorporating interdisciplinary knowledge: Graph structured data sometimes traces back to other disciplines such as biology, chemistry and social sciences. The interdisciplinary nature helps to leverage domain knowledge to solve specific problems, but it can also complicate model designs. For the case of molecular graph generation, the chemical constraints and the generation's objective function are often non-differentiable. Hence, gradient based training methods are out of the picture.

3) Ways to use GNNs: In [4], a general design to pipeline of GNNs is proposed. The following steps are necessary to obtain a graph-based deep learning framework:

a) Finding a graph structure: Based on the application in hand, we need to find out the underlying graph structure. There are two possibilities. First one is that we have an explicit graph structure, in the application such as social network, physical system or knowledge graph. The other possibility is that the underlying graph is implicit, and we need to build the graph from the task, such as obtaining a fully-connected "word" graph for text or obtaining a scene graph for an image. Then, we can obtain an optimal GNN model for the the graph we obtained either from explicit information or from the task.

b) Design a loss function: Based on the task in hand and the training setting, a loss function needs to be determined, the loss function can be node-level, edge-level or graph-level, depending on the training setting of supervised, semi-supervised or unsupervised learning.

c) Build model using computational modules: Finally, we need computational modules to build and train our model. Based on the definition provided in [4], we need a module to conduct convolution and recurrent operations to propagate information between nodes to capture the underlying feature and topological information. We need a sampling module, and we need a pooling module. With the combination of these modules a typical architecture of GNN model can be built.

III. Related Work

Some of the earliest success achieved on deep learning with graphs relied on finding proper ways to embed nodes into vectors using an encoder function [13]. One question arises from that definition is the *"What is a good representation?"*. We want these nodes embeddings to preserve interesting structures of the graphs. There are unsupervised graph representations learning algorithms like *node2vec* [14], *DeepWalk* [15] and LINE [16] which are trained prior to graph neural networks. These algorithms aimed to learn representative embeddings for nodes to preserve interesting structures of the graphs

Aforementioned algorithms inherently capture local similarities. Further studies find that Convolutional Graph Neural Networks (ConvGNNs) summarises local patches of the graphs and shows that neighbouring nodes tends to highly overlap [13]. Therefore, a ConvGNNs enforce similar features for neighbouring nodes by its nature without needing pre-training for node embeddings. This phenomenon was also mentioned in Text Graph Convolutional Network (Text GCN) [17]. Results of this paper on multiple benchmark data sets demonstrate that a vanilla Text GCN without any external word embeddings or knowledge outperforms state-of-the-art methods for text classification. On the other hand, Text GCN also learns predictive word and document embeddings jointly.

In [17] they evaluate Text GCN on two experimental tasks. First they seek the answer of whether their model achieve satisfactory results in text classification, even with limited labeled data and then they test whether their model learn predictive word and document embeddings. They compare their method with several state-of-art models. The suggested Text GCN may produce high text classification results and train predictive document and word embeddings, according to the experimental results. However, a major limitation of this study is that the GCN model is inherently transductive [17], in which test document nodes (without labels) are included in GCN training. Thus Text GCN could not quickly generate embeddings and make prediction for unseen test documents.

To improve the weakness of Text GCN in text classification task, [18] and [19] was mentioned in future work section of [17]. In [18] a novel algorithm Graph Attention Networks (GATs) was proposed. It was mentioned in [18] that GATs are new convolutional-style neural networks that operate on graph-structured data and use masked self-attentional layers. The graph attentional layer used in these networks is computationally efficient. Attentional layers do not require expensive matrix operations and they are parallelizable across all nodes in the graph. This structure allows for (implicitly) assigning different importance to different nodes within a neighborhood while dealing with different sized neighborhoods, and does not require knowing the entire graph structure. The experimental results yields that their attention-based models outperformed or matched state-of-the-art performance in four well-known node classification benchmarks, both transductive and inductive tasks [18].

Fast Graph Convolutional Neural Network [19] was also mentioned in the future work section of [17]. In [19] it was mentioned that, GCN in [20] represented as a useful graph model for semi-supervised learning. This model was created with the intention of being taught with both training and test data. Furthermore, for training with large, dense graphs, the recursive neighborhood expansion across layers faces time and memory issues. [19] interpret graph convolutions as integral transforms of embedding functions under probability measures to relax the condition of simultaneous availability of test data. As a result of this interpretation, Monte Carlo techniques may be used to consistently estimate the integrals, leading to a batched training scheme like FastGCN, which is proposed [19].

After further development on top of Text GCN, Simplifying Graph Convolutional Networks [21] was proposed to overcome unnecessary complexity and redundant computation in the previous work of Fast GCN and GATs. GCNs and their variants have received a lot of attention and have become the defacto methods for learning graph representations. GCNs are primarily inspired by modern deep learning methodologies, and as a result, they may inherit extra complexity and redundant processing. In [21] they eliminate the unnecessary complexity in this paper by reducing non-linearities one by one and collapsing weight matrices between layers. The resulting linear model is theoretically analyzed and shown to correspond to a fixed low-pass filter followed by a linear classifier in. The test results in [21] shows that these simplifications have no negative influence on accuracy in a wide range of downstream applications. Furthermore, the resulting model scales to bigger datasets, is intuitively interpretable, and outperforms FastGCN by up to two orders of magnitude.

References

[1] Kowsari, J. Meimandi, Heidarysafa, Mendu, Barnes, and Brown, "Text classification algorithms: A survey," *Information*, vol. 10, no. 4, p. 150, Apr. 2019, ISSN: 2078-2489. DOI: 10.3390/info10040150. [Online]. Available: http://dx.doi.org/10.3390/info10040150.

[2] Q. Li, H. Peng, J. Li, *et al.*, "A survey on text classification: From shallow to deep learning," *CoRR*, vol. abs/2008.00364, 2020. arXiv: 2008.00364. [Online]. Available: https://arxiv.org/abs/2008.00364.

[3] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning based text classification: A comprehensive review," *CoRR*, vol. abs/2004.03705, 2020. arXiv: 2004.03705. [Online]. Available: https://arxiv.org/abs/2004.03705.

[4] J. Zhou, G. Cui, S. Hu, *et al.*, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020, ISSN: 2666-6510. DOI: https://doi.org/10.1016/j.aiopen.2021.01.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666651021000012.

[5] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, Jan. 2021, ISSN: 2162-2388. DOI: 10.1109/tnnls.2020.2978386. [Online]. Available: http://dx.doi.org/10.1109/TNNLS.2020.2978386.

[6] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *CoRR*, vol. abs/1812.04202, 2018. arXiv: 1812.04202. [Online]. Available: http://arxiv.org/abs/1812.04202.

[7] L. Sun, J. Wang, P. S. Yu, and B. Li, "Adversarial attack and defense on graph data: A survey," *CoRR*, vol. abs/1812.10528, 2018. arXiv: 1812.10528. [Online]. Available: http://arxiv.org/abs/1812.10528.

[8] D. Cummings and M. Nassar, "Structured citation trend prediction using graph neural networks," *CoRR*, vol. abs/2104.02562,

2021. arXiv: 2104.02562. [Online]. Available: https://arxiv.org/abs/2104.02562.

[9] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. arXiv: 1810.04805. [Online]. Available: http://arxiv.org/abs/1810.04805.

[10] J. Howard and S. Ruder, "Fine-tuned language models for text classification," *CoRR*, vol. abs/1801.06146, 2018. arXiv: 1801.06146. [Online]. Available: http://arxiv.org/abs/1801.06146.

[11] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," arXiv preprint 1804.07461, 2018. [Online]. Available: https://gluebenchmark.com/.

[12] F. Barbieri, J. Camacho-Collados, L. Espinosa-Anke, and L. Neves, "TweetEval: Unified Benchmark and Comparative Evaluation for Tweet Classification," in *Proceedings of Findings of EMNLP*, 2020.

[13] P. Veličković, *Theoretical foundations of graph neural networks*, 2021. [Online]. Available: https://petar-v.com/talks/GNN-Wednesday.pdf.

[14] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," *CoRR*, vol. abs/1607.00653, 2016. arXiv: 1607.00653. [Online]. Available: http://arxiv.org/abs/1607.00653.

[15] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk," *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, Aug. 2014. DOI: 10.1145/2623330.2623732. [Online]. Available: http://dx.doi.org/10.1145/2623330.2623732.

[16] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line," *Proceedings of the 24th International Conference on World Wide Web*, May 2015. DOI: 10.1145/2736277.2741093. [Online]. Available: http://dx.doi.org/10.1145/2736277.2741093.

[17] L. Yao, C. Mao, and Y. Luo, *Graph convolutional networks for text classification*, 2018. arXiv: 1809.05679 [cs.CL].

[18] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, *Graph attention networks*, 2018. arXiv: 1710.10903 [stat.ML].

[19] J. Chen, T. Ma, and C. Xiao, "Fastgcn: Fast learning with graph convolutional networks via importance sampling," *CoRR*, vol. abs/1801.10247, 2018. arXiv: 1801.10247. [Online]. Available: http://arxiv.org/abs/1801.10247.

[20] T. N. Kipf and M. Welling, *Semi-supervised classification with graph convolutional networks*, 2017. arXiv: 1609.02907 [cs.LG].

[21] F. Wu, T. Zhang, A. H. S. Jr., C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," *CoRR*, vol. abs/1902.07153, 2019. arXiv: 1902.07153. [Online]. Available: http://arxiv.org/abs/1902.07153.