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*—Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.**

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

I. Introduction

- Introduction to Text Classification task, overview of the previous approaches, in Section II-A.
- Introduction to Graph Neural Networks (GNNs), what is GNN, why they are being utilized, in Section II-B.
- GNN in NLP, more specifically in Text Classification in Section III.

II. Previous Work

A. Text Classification

In [1], *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 [2] 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 [2], 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 [2] 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 [1], 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 [4] variations and fine tuning pre-trained language models [5], 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 [1], [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 [6], TweetEval [7], 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 [8]–[11].

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 [9]:

- 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 [9]. Some of the main challenges can be categorized as follows [10]:

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 [8], 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 [8], 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

[12]–[16]

References

[1] 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.

[2] 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.

[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. 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.

[5] 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.

[6] 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/.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

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

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

[14] H. Peng, J. Li, Q. Gong, *et al.*, *Hierarchical taxonomy-aware and attentional graph capsule rcnns for large-scale multi-label text classification*, 2019. arXiv: 1906.04898 [cs.IR].

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

[16] Z. Guo, Y. Zhang, and W. Lu, "Attention guided graph convolutional networks for relation extraction," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 241–251. DOI: 10.18653/v1/P19-1024. [Online]. Available: https://aclanthology.org/P19-1024.