Spatial-Temporal Aware Disaster Topic Recognition in Social Media Text Using SageMaker

Zheng He[1], Zhifei Luo[2], Lin Li[2]
[1]Deloitte Consulting Co., Ltd., Shanghai, China
[2]Wuhan University of Technology, Wuhan, Hubei, China

Abstract: With the popularization and widespread use of social media, a large amount of information is shared and disseminated by users on the platforms, including disaster-related text messages. The disaster information contained in these social media text messages is of great value for emergency response, disaster research, and the understanding of public opinion. In the task of text classification of social media disaster topics, previous approaches generally only semantically analyze the text itself to determine its relevance to the disaster, and this paper considers a new perspective that combines the spatio-temporal attributes of social media texts and textual information to complete the text classification task. Based on the above idea, this paper constructs a heterogeneous graph on the dataset using the temporal and spatial attributes of blog posts, initializes the blog nodes using BERT features, and learns the blog features through relational graph convolution operation. Experiments on the SageMaker platform use BERT as a baseline model, while blog post features are initialized using this model. The experimental results show that the spatio-temporal model achieves better performance compared to the baseline model.

Keywords: Disaster Detection; Social Media; Graph Convolutional Networks; Sagemaker; Text Categorization

1. Introduction

With the popularity of social media and the rapid growth in the number of users, social media has become an important platform for people to obtain information, share opinions and express their emotions. From daily life trivia to major events, a large number of disaster-related texts have emerged on social media. These texts include reports of disaster events, real-time descriptions from eyewitnesses, messages for help and appeals for assistance. As a source of disaster information, social media texts are uniquely real-time, widespread, and diverse. They reflect the on-the-spot situation at the site of a disaster, the reactions and needs of the population, and become an important reference for disaster management and emergency response.

However, the massive nature of social media texts poses a huge challenge for accurate categorization and analysis. Hundreds of millions of social media users post a variety of disaster-related content on the platform every day, which makes manual processing and analysis impractical. Therefore, there is an urgent need to apply automated methods to effectively process and utilize this massive information resource.

The study of categorizing social media disaster event texts has important practical and theoretical implications. At the practical level, by accurately categorizing social media texts, it is possible to better understand the extent of the relationship between the texts and disasters. This helps to identify which texts are closely related to disasters, thus providing more accurate information and guidance to support disaster management, emergency response and assistance decisions. Ultimately, it helps to reduce losses and injuries caused by disasters, protect people's lives and property, and improve the resilience of society. At the theoretical level, categorizing social media disaster texts can provide more data resources for disaster research. Researchers can analyze the disaster information contained in the texts and study in depth the impact, dissemination mechanism and social dynamics of disaster events. This will provide useful support for further development and theory construction in the field of disaster research.

2. Methodology

2.1 Framework

The model constructed in this paper uses the spatio-temporal information of blog posts, i.e., the time when the blog post is published and the geographic location of the user at the time the blog post is published, to provide contextual information on the spatio-temporal aspects of blog post categorization. Disaster-related blog posts are often associated with disaster events, and temporal and spatial information are important attributes of disaster events. In this paper, we utilize blog post information and spatio-temporal information to construct a graph structure and use relational graph convolutional networks on it for blog post feature learning. The spatio-temporal model framework is shown in Fig. 1.

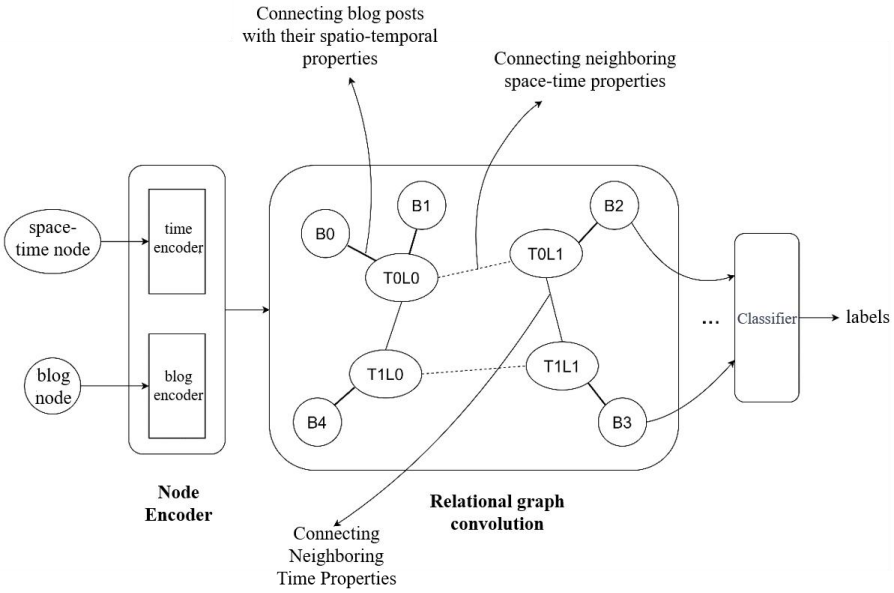

Figure 1. Spatio-Temporal Modeling Framework

B0 in the spatio-temporal modeling framework diagram denotes the blog post numbered 0, and T0L0 denotes the combination of time numbered 0 and space numbered 0.

2.2 Node Coding Module

The node encoding module consists of two parts: the blog post node encoder and the spatio-temporal node encoder. Blog node encoder[1,2] Taking the text as input, we extract the hidden representation corresponding to the "[CLS]" tag through the fine-tuned BERT[3,4] model, extracts the hidden representation corresponding to the "[CLS]" tag, i.e., a sentence-level encoded representation of length 768, as output. The spatio-temporal node encoder, on the other hand, represents each spatio-temporal node uniformly as an all-zero feature vector of the same length as the output of the blog post node encoder.

2.3 Relational Graph Convolution Module

The spatio-temporal graph of the Relational Graph Convolution module contains 2 types of nodes, blog post nodes and spatio-temporal nodes. A blog post node is a single blog post as a whole as a node in the spatio-temporal graph. Spatio-temporal node is a node abstracted from spatio-temporal information, time information is in days, generating time information points within a defined time range, spatial information is in provinces, generating spatial information points based on administrative divisions, and generating spatio-temporal nodes by performing Cartesian product operation on time information points and spatial information points.

Spatio-temporal graphs have 4 types of edges, namely temporal edges, spatial edges, blog post to spatio-temporal edges and spatio-temporal to blog post edges.

2.3.1 Time side

Temporal edges are space-time nodes with the same spatial attributes and different temporal attributes, which are connected by temporal edges in chronological order to form a chain structure. For example, there are three

space-time nodes, in chronological order from near to far, respectively, "time 1-space A", "time 2-space A" and "time 3-space A". Then the nodes "Time 1-Space A" and "Time 2-Space A" are connected by a temporal edge (bidirectional), and the nodes "Time 2-Space A" and "Time 3-Space A" are connected by a temporal edge (bidirectional). "time2-spaceA" node and "time3-spaceA" node are connected by a temporal edge (bi-directional).

In the figure, temporal edges are able to communicate information about blog posts in neighboring time ranges and consider the evolution of blog posts at different points in time, thus obtaining more accurate and comprehensive temporal contextual information to capture disaster events within a time period. Temporal edges allow the model to take into account temporal continuity, which leads to a better understanding of the connection between social media texts and disaster events and improves the model's performance on the task of categorizing social media disaster event texts. The following paragraph is excerpted from the NDRN as an example of the temporal continuum of disasters.

"October 2-6, China's first cold wave weather process in the second half of this year, from north to south affecting most areas, the central and eastern regions of the drastic cooling, Shanxi, Henan and other places in the occurrence of low-temperature freezing disaster, resulting in potato, corn and other parts of the crop suffered frost damage to reduce production or crop failure, crop damage to an area of 17.4 kilo-hectares. Direct economic losses of 420 million yuan."

2.3.2 Space edge

Spatial edges connect spatial and temporal nodes with the same temporal attributes and different spatial attributes, which are connected by spatial edges based on the neighboring relationship between provinces to form a mesh structure. Take Hubei province as an example, there are three spatio-temporal nodes, namely, "Time 1-Hubei province", "Time 1-Anhui province" and "Time 1-Chongqing city", according to the neighboring information between Hubei and other provinces, "Time 1-Hubei province" is connected to "Time 1-Chongqing city". According to the neighboring information between Hubei Province and other provinces, the nodes of "Time1-Hubei Province" and "Time1-Anhui Province" are connected by a spatial edge (bidirectional), and the nodes of "Time1-Hubei Province" and "Time1-Chongqing Municipality" are connected by a spatial edge (bidirectional). The node "Time1-Hubei" and the node "Time1-Anhui" are connected by a spatial edge (bidirectional).

The use of spatial edges to connect neighboring spatial nodes enables the communication of information about blog posts in neighboring spatial scales to capture disaster events in one geographic scale. The introduction of spatial edges enables the model to take into account the geospatial distribution and correlation of blog posts, further enhancing the understanding of the correlation between social media texts and disaster events, capturing the propagation and impact of disaster events in the geographic dimension, and providing a more discriminating feature representation for the task of categorizing social media disaster event texts. The following paragraph is excerpted from the NDRN as an example of the spatial continuum of disasters.

"The total number of fires starts declined, down more than 1,200 compared with the average of the same period in the past five years; the distribution area is concentrated, affected by the persistent high temperature and little rain in the south, Hunan, Guangxi, Jiangxi, Hubei, Chongqing and other southern provinces are the most frequent areas of forest fires."

2.3.3 Blogging to the edge of time and space

Each blog post contains temporal and spatial information, and at the same time there exists a unique corresponding spatio-temporal node in the set of spatio-temporal nodes, based on this relationship, the edge pointing to its corresponding spatio-temporal node from a blog post node is the blog post to spatio-temporal edge. By using the blog-to-space-time edge, a spatio-temporal node can aggregate the information of all the blogs it corresponds to, which means that the spatio-temporal node is able to comprehensively consider the characteristics of multiple blogs on the space-time, thus providing more comprehensive and rich contextual information.

2.3.4 Time and space to the side of the blog

A space-time to blog post edge is the reverse of a blog post to space-time edge, i.e., it is pointed by a space-time node to a blog post node corresponding to it. The spatio-temporal node to blog post node edge connection realizes the transfer of spatio-temporal information to the blog post node, so that the blog post node can obtain the information of the spatio-temporal where it is located and update its own characteristics. Through the spatio-temporal to blog post edges, the model is able to better understand the correlation between blog posts and disaster events, capture the semantics of blog posts in a specific spatio-temporal range, and further improve the performance of the model.

In this paper, we use R-GCN[5,6] to learn blog post features on the above heterogeneous graphs.R-GCN (Relational Graph Convolutional Network) is a graph neural network based[7-9] R-GCN (Relational Graph Convolutional Network) is a model based on graph neural networks for processing graph data containing multiple relationship types. Since R-GCN considers relationship types in the graph structure, it is better able to capture the association information between nodes. Traditional graph neural networks[10] usually assumes that all nodes have the same relationship with each other, R-GCN introduces relationship-specific weight parameters that give each relationship type a different influence. In this way, R-GCN is able to adjust the information transfer between nodes according to the specific type of relationship when learning node features. Through multi-layer graph convolution operation, R-GCN is able to gradually aggregate the neighbor information of nodes to generate richer node features. The propagation function is shown in (1).

$$h_i^{(l+1)} = \sigma(W_0^{(l)} h_i^{(l)} + \sum_{r\in R} \sum_{u\in U_{i,r}} \frac{1}{z_{i,r}} W_r^{(l)} h_u^{(l)}) \quad (1)$$

$h_i^{(l)}$ indicates that the first l node in the layer i feature of the node, and $h_i^{(0)}$ is the initial feature of the node; $U_{i,r}$ denotes the node i under the relationship type r set of neighboring nodes under the relationship type; $z_{i,r}$ It is used for the node i under the relation type r under the regularization; $W_r^{(l)}$ denotes the set of neighboring nodes in the first l relationship type in the layer r under the weight parameter, the $W_0^{(l)}$ denotes the weight parameter under the l weight parameter under the self-loop relationship in the layer.

2.4 Classification Module

The classification module first performs a linear transformation on the feature part of the blog post output by the relational graph convolution module, transforming the feature vector of length 768 into a vector of length 3. Afterwards, the linear transformation result is passed through the nonlinear activation function ReLU to increase the nonlinear ability of the classifier, and finally the Softmax function is applied to the result after activation of ReLU to obtain the probability distribution of the labels, and the label with the maximum probability is selected of the label is selected as the final classification result.

2.5 Training Algorithm

In this paper, the spatio-temporal model is trained using full batch gradient descent technique and the training algorithm is shown in Algorithm 1:

Input: θ , initial model parameters; G , spatio-temporal map; $\{(x_n, t_n)\}_{n=1}^{N_{data}}$, text dataset

Output:$\theta^{'}$, trained model parameters.

Hyperparameters: $N_{epochs} \in \mathbb{N}$, number of training rounds;$\eta \in (0, \infty)$, learning rate.

Line 1, for $j = 1, 2, \cdots, N_{epochs}$ do

Line 2, $H(\theta) \longleftarrow \text{BertRgcnClassifier}(G, x_n|\theta)$

Line 3, $\text{loss}(\theta) = -\sum_{i\in Y} t_i \ln H_i(\theta)$ /* **Y** is the set of nodes whose labels are known */

Line 4, $\theta \longleftarrow \theta - \eta \cdot \nabla \text{loss}(\theta)$

Line 5, end

Line 6, return $\theta' = \theta$

3. Experimental

3.1 Introduction to the Experimental Platform

The SageMaker platform is a fully hosted machine learning platform provided by Amazon AWS. SageMaker provides a computing infrastructure for automated build and maintenance, a rich set of powerful machine learning tools, and clear workflows that enable developers to easily build, train, and deploy various types of machine learning models.

In this paper, the task of social media disaster topic text categorization can be accomplished

based on SageMaker platform. The task uses Relational Graph Convolutional R-GCN (Relational Graph Convolutional Network) model for feature learning and chooses to use DGL (Deep Graph Library) library to process graph data, and Amazon SageMaker provides support for DGL. Among them R-GCN is a graph neural network-based model for processing graph data containing multiple relationship types; DGL is a high-performance library designed for graph neural networks, which provides a rich set of graph operations and algorithms that can help us perform efficient computation and training on large-scale graph data.

3.2 Experimental Steps

(1) Upload the dataset: create a Notebook instance (SageMaker-supported fully managed Jupyter Notebook in the cloud), generate a SageMaker session and get_execution_role, create an S3 bucket, and upload the local text dataset to the corresponding location in S3 using SageMaker's encapsulated upload _data interface encapsulated by SageMaker to upload the local text dataset to the corresponding location in S3.

(2) Upload model files and training scripts: upload the model file (model.py) that defines the PyTorch-based text categorization model, and the script file (train.py) that defines the training process.

(3) Define Estimator object: generate an Estimator object (Estimator is a high-level abstraction provided by Amazon SageMaker to simplify the process of training and deploying machine learning models), set the training parameters, such as the training entry point, i.e., the path to train.py, the type of the training instance, train_ instance_type, PyTorch framework version and python version (specifying these two parameters allows you to directly use the DL Container provided by Amazon SageMaker to train the DGL model), and so on.

(4) Train the model: use the IntegerParameter module provided by SageMaker to set the hyperparameter adjustment range of the model, define a HyperparameterTuner object (HyperparameterTuner is a tool provided by SageMaker for automatically adjusting the hyperparameters of machine learning models), pass in parameters such as Estimator object and call the HyperparameterTuner interface to train the model; call the fit interface of the HyperparameterTuner object to train the model.), pass in the Estimator object, hyperparameter tuning range, and other parameters, and call the HyperparameterTuner object's fit interface for model training.

(5) Deploy the model: Create a PyTorchModel based on the trained Estimator object (PyTorchModel is the object used in SageMaker to deploy and execute PyTorch models), call the deploy interface of the PyTorchModel to deploy the model, and then call the predict interface of the PyTorchModel to complete the evaluation of the model. predict interface to evaluate the model.

3.3 Experimental Data and Parameter Settings

In this paper, the dataset is divided into training set, validation set and test set in the ratio of 8:1:1. There are 14858 blog posts in the training set, 1857 blog posts in the validation set and 1858 blog posts in the test set.

3.3.1 Text length distribution

The training set has 14,858 blog posts, with an average of 39 words per blog post, a shortest sentence length of 3, and a longest sentence length of 197, in which less than 75% of the data is within 51, less than 50% is within 28, and less than 25% is within 17. The test set has 1858 blog posts with an average of 39 words per blog post, with a shortest sentence length of 3 and a longest sentence length of 206, in which less than 75% of the data is within 51, less than 50% is within 28, and less than 25% is within 17. The metrics of the training and test sets are very close from the data description point of view. The frequency histogram of text length distribution is shown in Figure 2.

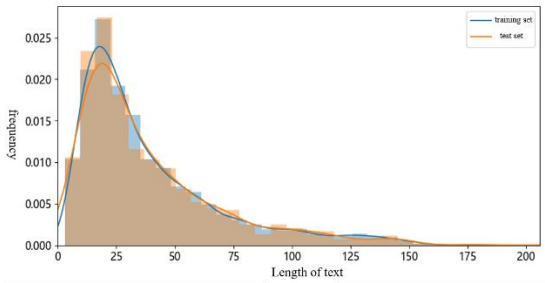

Figure 2. Frequency Histogram of Text Length Distribution

The distribution frequency histogram shows that the training and test sets are

approximately identically distributed. The P-value was calculated using the stats. ks_2samp function of the scipy module, and the result was 0.97, which is greater than the specified significance level of 0.05. It is therefore concluded that the training set and the test set are identically distributed with respect to the text lengths, and that they can be considered to be sampled from the same data distribution.

3.3.2 Text label distribution

The training set has a total of 14,858 blog posts, of which there are 7,935 high relevance texts, accounting for 53.4% of the total number of training sets, 734 medium relevance texts, accounting for 4.9% of the total number of training sets, and 6,189 medium relevance texts, accounting for 41.7% of the total number of training sets. There are 1858 blog posts in the test set, of which 996 are high relevance texts, accounting for 53.6% of the total test set, 106 are medium relevance texts, accounting for 5.7% of the total test set, and 756 are medium relevance texts, accounting for 40.7% of the total test set. From the data description point of view, the metrics of the training and test sets are very close.

The P-value was calculated using the stats. ks_2samp function of the scipy module, and the result was 0.99, which is greater than the specified significance level of 0.05. It is therefore concluded that the training and test sets are identically distributed with respect to the text labels, and that they can be thought of as sampling from the same data distribution.

For the spatio-temporal model, this paper uses the Adam optimizer, sets the initial learning rate to 5e-5, sets the number of training rounds to 40, and employs full batch gradient descent. In this paper, three models with different numbers of R-GCN layers are constructed, which are 2-layer, 4-layer, and 6-layer. The hidden layer size decreases exponentially, taking the 2-layer model as an example, the input feature size is 768, the hidden layer size of layer 1 is 384, and the hidden layer size of layer 2 is 192.

In this paper, we use BERT as the baseline model, also set the optimizer to Adam, set the initial learning rate to 1e-6, set the number of training rounds to 20, and the batch size to 4.

3.4 Experimental Results and Analysis

The experimental results are shown in Table 1. The BERT model first sorts the input text and adds special tokens to indicate the beginning, end and interval of the sentence, then, after the embedding layer, each word is converted into a corresponding word vector, which is then passed through multiple layers of the Transformer[11] encoder, and finally, the feature vectors are extracted from the output of the last layer of the Transformer, which is fed into the classifier for classification. The experiments show that the accuracy of BERT on the test set is 86.3% with a running time of 6.8e-4 s. Among the spatio-temporal models, the 4-layer model achieves the highest accuracy with a running time of 5.7e-2 s. The BERT is a spatial and temporal model with a running time of 5.7es.

Table 1. Model Results

Mould	Accuracy	Running time (seconds)
BERT	86.3	6.8e-4
Space-time model (2 layers)	87.6	4.4e-2
Space-time model (4 layers)	88.5	5.7e-2
Space-time model (6 layers)	87.8	6.3e-2

Comparing with the baseline model, the 4-layer spatio-temporal model showed a 2.2% improvement in accuracy on the test set, and the less accurate 2-layer and 6-layer models still showed improvement compared to the baseline model. This suggests that spatio-temporal information provides assistance for social media disaster event text categorization. On the one hand, through the linkage between blog posts and spatio-temporal attributes, blog posts can obtain contextual information about their spatio-temporal location; on the other hand, through the linkage between spatio-temporal attributes and spatio-temporal attributes, the model can capture information about disasters within a spatio-temporal range. Overall, the spatio-temporal model introducing spatio-temporal information has a better grasp of the characteristics of social media disaster information texts.

In terms of running time, the running time of the spatiotemporal model becomes longer with the increase of the number of R-GCN layers, and overall, the running time of the spatiotemporal model is much longer than that

of the BERT model, and the difference in the time between the two is the time spent by the spatiotemporal model to carry out the relational graph convolution operation in order to fuse the spatiotemporal information. Although there is an order of magnitude difference between the running time of the spatiotemporal model and the baseline model, the time spent on performing the relational graph convolution operation is still within an acceptable range when considering the absolute value of the running time of the spatiotemporal model and the size of the test set. The model accuracy line graph is shown in Figure 3.

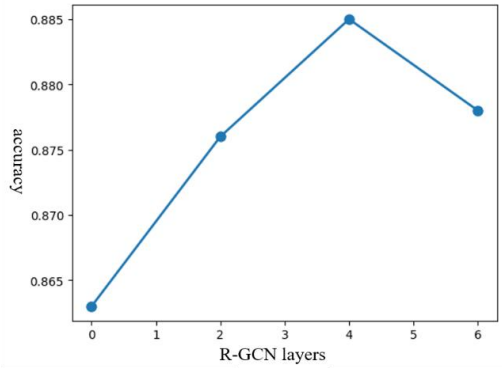

Figure 3. Line Graph of Model Accuracy

The baseline model is equivalent to the layer 0 spatio-temporal model. From the line graph, the 2-layer model shows a 1.3% increase in accuracy compared to the 0-layer, thanks to the fact that the 2-layer model can capture the contextual information of the spatio-temporal space in which a blog post is located.The 4-layer model shows a 0.9% increase in accuracy compared to the 2-layer, due to the fact that, on top of the 2-layer model, the 4-layer model can capture the features of the neighboring spatio-temporal space of a blog post, which provides it with a more complete set of contextual information. The 6-layer model There is a 0.7% decrease in accuracy compared to the 4-layer, which is due to the fact that the 6-layer model incorporates a larger range of neighboring spatio-temporal features than the 4-layer model, but due to the large range, the more distant spatio-temporal features instead become noise in the contextual information, which makes the quality of the contextual information decrease.

4. Conclusion

This paper aims to address the challenge of social media disaster event text classification and achieves accurate classification by constructing a heterogeneous spatio-temporal graph and employing the R-GCN model, with the spatio-temporal model achieving a 2.2% accuracy improvement on the test set relative to the baseline model. This indicates that the features of social media disaster event texts can be more accurately captured by introducing spatio-temporal information to improve the classification performance.

In this paper, the spatio-temporal attributes of blog posts are mined as additional information to help with the text categorization task. Blog posts, as texts on social media platforms, also have other attributes, such as comments under the blog post, the retweeting of the blog post, and information about the users who have participated in the blog post, etc. Whether integrating these attributes into a graph to provide more comprehensive contextual information about the blog post will bring about an improvement in the performance of the model can also be explored in a further Direction.

References

[1] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 2012, 25.

[2] Yoon Kim. Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).2014:1746-1751.

[3] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[4] Qu C, Yang L, Qiu M, et al. BERT with history answer embedding for conversational question answering. Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval. 2019: 1133-1136.

[5] Schlichtkrull M, Kipf T N, Bloem P, et al. Modeling relational data with graph convolutional networks. The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings 15. Springer International Publishing, 2018: 593-607.

[6] Mehta N, Pacheco M L, Goldwasser D. Tackling fake news detection by continually improving social context representations using graph neural networks. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. 2022: 1363-1380.

[7] Yao L, Mao C, Luo Y. Graph convolutional networks for text classification. Proceedings of the AAAI conference on artificial intelligence. 2019, 33 (01): 7370-7377.

[8] Wu T, Liu Q, Cao Y, et al. Continual Graph Convolutional Network for Text Classification. arXiv preprint arXiv:2304.04152, 2023.

[9] Huang L, Ma D, Li S, et al. Text level graph neural network for text classification. arXiv preprint arXiv:1910.02356, 2019.

[10] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907, 2016.

[11] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. Advances in neural information processing systems, 2017, 30.

