Graph Recurrent Neural Network for Text Classification

Anonymous ACL submission

Abstract

Graph Neural Networks(GNNs) application to text classification is currently one of the most popular fields. Most GNNs-based models only focus on the interaction of words in the document, whereas the word order is ignored, and the related semantic information is lost. In addition, when the graph density increases, the word nodes become over-smooth. As a result, the semantic information of the document is destroyed. In this paper, TextGRNN, a text classification method based on GNN is proposed to solve the above problems. First, our proposed model constructs the document-level graph via Visibility Graph, in which the graph density is restrained, and updates the word representations by GNN. Then the TextGRNN model utilizes Bi-LSTM that can recognize word order to learn the semantic information of the document. Finally, the attention mechanism is used to highlight the essential words. Numerous experiments on three benchmark datasets demonstrate that our model is preferable to state-of-the-art text classification methods.

1 Introduction

Text classification is an important research area in natural language processing (NLP). It is necessary for many practical scenarios, such as news filtering, spam detection, sentiment analysis and author attribution. Traditional text classification methods focus on algorithms and rely on hand-made features like n-gram (Brown et al., 1992) and TF-IDF (Wu et al., 2008), resulting in low efficiency.

As deep technology progresses, more deep learning models are employed for text classification and show better results than traditional models. Most of the deep learning models are extended based on convolutional neural networks (Kim, 2014) or recurrent neural networks (Mikolov et al., 2010), for instance, TextCNN (Kim, 2014), TextRNN (Mikolov et al., 2010) and TextRCNN (Lai et al., 2015). However, it is challenging to achieve information interaction for some words if there is a long distance between them. Therefore, Graph Neural Networks (GNNs) are used to solve this problem.

GNN is introduced in (Gori et al., 2005) and extensively applied in NLP (Nikolentzos et al., 2020). In text classification, the corpus-level models (Yao et al., 2019; Wu et al., 2019; Wang et al., 2020) construct the graph over the entire corpus, where the word nodes and the edges are fixed globally. These models fail inductive learning, and it is difficult to represent new documents with new structures and words. The document-level models (Ding et al., 2020; Xie et al., 2021; Zhang et al., 2020) build graphs for each document by Word Co-occurrence, aiming to deal with the problem of inductive learning. However, as the graph density becomes more extensive, the nodes become over-smooth. thus the semantic information of the documents is corrupted. To prevent this problem, the sliding window size is set small(Zhang et al., 2020), which makes it is hard for words to interact with more words. In addition, some GNN-based models no longer consider the text as a sequence, which would lose semantic information about word order. Nikolentzos et al. (2020) establish a directed graph to represent text flow, but after multiple iterations of information transmission, the word nodes became over-smooth, and it is also difficult to convey semantic information.

To solve these issues, in this work we propose a new inductive learning model(TextGRNN), which contains the GNN layer, the Bi-LSTM layer and the attention layer. The document-level graph is built by Visibility Graph, in which a document is translated into time series. The series is split up into sliding fragments by the sliding window. Our proposed model establishes corresponding visibility sub-graphs (Stephen et al., 2015) for sliding chips and consequently build a visibility graph for the document through the same word nodes of visibility sub-graphs. After propagating the embedding of word nodes to their neighbors through the GNN layer, word nodes are input into the Bi-LSTM layer that perceives word order, in order to extract semantic information. Finally, the attention layer highlights the essential words of documents. To sum up, our contribution is threefold:

- We propose a new method to construct the graph of a document, named Visibility Graph, which reduces the density of the graph in the case of large-size sliding windows. As a result, word nodes can avoid becoming over-smooth and interact with more word nodes.
- Our model enables long-distance word interaction as well as awareness of word order, which powerfully expresses the semantic information of the document.
- Our approach achieves state-of-the-art results on several benchmark datasets compared with the baselines.

2 Related Work

Recently, GNNs have been widely applied to text classification. Yao et al. (2019) take word-word and word-document relations into account and build a graph on the entire corpus. Wang et al. (2020) enrich the represents of the corpus-level graph by considering topic-level nodes. Huang et al. (2019) create the text-level graph, whose edges of each team of words are also globally fixed.

The above models with globally fixed nodes and edges are difficult to handle new documents, so some studies focus on inductive representation. Ding et al. (2020) propose a hypergrah for every document and capture highlights text representation through dual attention mechanism. Zhang et al. (2020) utilize the TextING for text classification, which can represent new words that do not appear in the training. Xie et al. (2021) blend a topic model in variational graph-auto-encoder to reflect the relevant semantic information. However, none of the these models consider the word order, which leads to the loss of relevant semantic information.

We notice that Liu et al. (2020) design the TensorGCN that contains the graph with semantic information based on word order, but the model still works on the whole corpus, which implies that TensorGCN is incapable of inductive learning.

Some researchers study the text from the perspective of time series. Vieira et al. (2018) measure the robustness of sentence length with six criteria, where a sentence length is portrayed as a data point of time series. Word lengths and word frequencies are also used to map the document into time series, which can quantify complexity in written texts (Ausloos, 2012). Visibility Graph is a network method that can analyze the time series. Stephen et al. (2015) construct visibility sub-graphs for the time series to provide rich information conducive to short-term and long-term forecasts.

3 Method

3.1 Visibility Graph

Step 1 A document is denoted as

$$\begin{aligned} &doc : \{(w_1, l_1), (w_2, l_2), (w_3, l_3),\\ &\quad \ldots, (w_j, l_j), \ldots, (w_n, l_n)\}, \end{aligned} \tag{1}$$

where w_j, n and l_j, respectively, are the j-th word, the number of words in the document and the word length of the w_j.

Step 2 We fix a window of size s to slide over doc, and the generated sliding fragment is shown as follows:

$$\begin{aligned} &doc_q = \{(w_q, l_q), (w_{q+1}, l_{q+1}),\\ &\quad \ldots, (w_{q+s-1}, l_{q+s-1})\},\\ &q = 1, 2, \ldots, n-s+1. \end{aligned} \tag{2}$$

Step 3 If w_{k_1}, w_{k_2} and w_{k_3} satisfy the criterion:

$$l_{k_2} \le l_{k_3} + (l_{k_1} - l_{k_2}) \cdot (\frac{k_3 - k_2}{k_3 - k_1}), \tag{3}$$

$$k_1, k_2, k_3 \in \{q, q+1, \ldots, q+s-1\}, \tag{4}$$

$$k_1 < k_2 < k_3, \tag{5}$$

w_{k_1} can see w_{k_3}, and there is an edge between them. Thus the doc_q can be mapped as a visibility sub-graph.

Step 4 Every sliding fragment has a corresponding visibility sub-graph. We represent unique words as nodes. Thus, these Visibility Sub-graphs can be connected to construct the Visibility Graph for doc through the same words. The document-level graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are a set of nodes and edges, respectively.

The document is preprocessed by tokenization and removal of stop words (Rousseau et al., 2015). The word features are used to initialize the nodes' embedding, denoted as $\mathbf{v} \in \mathbb{R}^{|\mathcal{V}| \times d}$, where d is the embedding dimension.

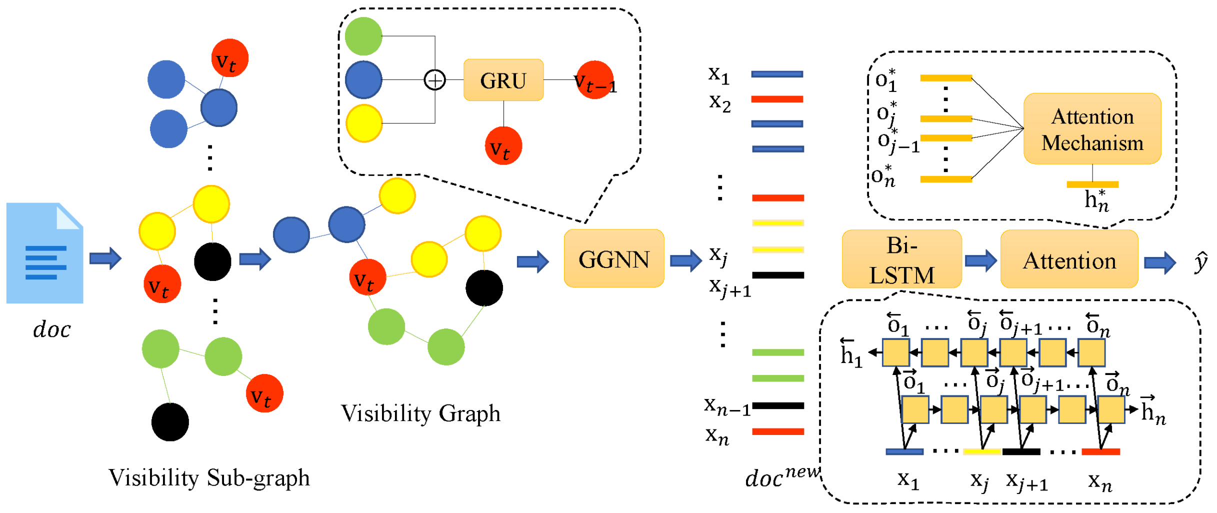

Figure 1: The architecture of TextGRNN. First, the graph of the document is built by Visibility Graph, and the word nodes are updated with GGNN. Then, Bi-LSTM captures semantic information based on word order. Finally, The attention mechanism highlights the essential words in the document.

3.2 Graph-based Word Interaction

On each graph, the embedding of word nodes can be updated by Gated Graph Neural Network (Li et al., 2016) as the following formulates:

$$\mathbf{a}_t = \mathbf{A}\mathbf{v}_{t-1}\mathbf{W}_a, \tag{6}$$

$$\mathbf{z}_t = \sigma(\mathbf{W}_z\mathbf{a}_t + \mathbf{U}_z\mathbf{v}_{t-1} + \mathbf{b}_z), \tag{7}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r\mathbf{a}_t + \mathbf{U}_r\mathbf{v}_{t-1} + \mathbf{b}_r), \tag{8}$$

$$\widetilde{\mathbf{v}}_t = \tanh(\mathbf{W}_v\mathbf{a}_t + \mathbf{U}_v(\mathbf{r}_t \odot \mathbf{v}_{t-1}) + \mathbf{b}_v), \tag{9}$$

$$\mathbf{v}_t = \widetilde{\mathbf{v}}_t \odot \mathbf{z}_t + \mathbf{v}_{t-1} \odot (1 - \mathbf{z}_t), \tag{10}$$

where $\mathbf{a}$ is the information of the current node's adjacent neighbours, $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the adjacency matrix, σ is the sigmoid function. All $\mathbf{W}$ and $\mathbf{U}$ are trainable weights, and $\mathbf{b}$ is the bias. $\mathbf{z}$ and $\mathbf{r}$ are the update and reset gates, which control the amount of information from the current node's previous step (in $\mathbf{v_{t-1}}$) and the current node's neighbors should contribute to the current node. t means that such update operates t times and nodes can achieve the information from their t-order neighbors.

3.3 Bi-LSTM

After updating the word nodes, the graph is converted to doc^{new}:

$$doc^{new} : \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_j, ..., \mathbf{x}_n\}, \tag{11}$$

where $\mathbf{x}_j$ is the updated embedding of the unique node corresponding to w_j.

LSTM has the latency capability to capture the semantic information represented by the word order(Iacobacci and Navigli, 2019). We use Bi-directional Long Short-Term Memory (Bi-LSTM) to learn the semantic information of the doc^{new}. The LSTM transition equations are as follows:

$$\mathbf{p}_j = \sigma(\mathbf{W}_p\mathbf{h}_{j-1} + \mathbf{U}_p\mathbf{x}_j + \mathbf{b}_p), \tag{12}$$

$$\mathbf{f}_j = \sigma(\mathbf{W}_f\mathbf{h}_{j-1} + \mathbf{U}_f\mathbf{x}_j + \mathbf{b}_f), \tag{13}$$

$$\mathbf{o}_j = \sigma(\mathbf{W}_o\mathbf{h}_{j-1} + \mathbf{U}_o\mathbf{x}_j + \mathbf{b}_o), \tag{14}$$

$$\widetilde{\mathbf{c}}_j = \tanh(\mathbf{W}_c\mathbf{h}_{j-1} + \mathbf{U}_c\mathbf{x}_j + \mathbf{b}_c), \tag{15}$$

$$\mathbf{c}_j = \mathbf{f}_j \odot \mathbf{c}_{j-1} + \mathbf{p}_j \odot \widetilde{\mathbf{c}}_j, \tag{16}$$

$$\mathbf{h}_j = \mathbf{o}_j \odot \tanh(\mathbf{c}_t), \tag{17}$$

where $\mathbf{f}$, $\mathbf{p}$ and $\mathbf{o}$ are forget gate, input gate and output gate, respectively. At the j-th time step, the memory $\mathbf{c}_j$ and hidden state $\mathbf{h}_j$ are updated as Eq.(12)$\sim$(15).

Bi-LSTM (Zhou et al., 2016) can acquire information from both the past and future of the document. The output of the Bi-LSTM are as follows:

$$\mathbf{output} = \{\mathbf{o}_1^*, \mathbf{o}_2^*, \mathbf{o}_3^*, ..., \mathbf{o}_j^*, ..., \mathbf{o}_n^*\}, \tag{18}$$

$$\mathbf{o}_j^* = \overrightarrow{\mathbf{o}_j} + \overleftarrow{\mathbf{o}_j}, \tag{19}$$

$$\mathbf{h}^* = \overrightarrow{\mathbf{h}_n} + \overleftarrow{\mathbf{h}_1}, \tag{20}$$

where $\overrightarrow{\mathbf{o}_j}$ and $\overrightarrow{\mathbf{h}_n}$ are the output of j-th step and last hidden state from the forward LSTM layer, $\overleftarrow{\mathbf{o}_j}$ and $\overleftarrow{\mathbf{h}_1}$ are the output of j-th step and last hidden state from the backward LSTM layer, respectively.

3.4 Attention Mechanism

To highlight essential words in the document, TextGRNN uses the attention mechanism (Zhou et al., 2016) to assign higher weights to significant terms:

$$\mathbf{e}_j = \tanh(\mathbf{h}^{*\mathrm{T}}\mathbf{W}_o^h\mathbf{o}_j^*), \tag{21}$$

$$\alpha_j = \frac{exp(\mathbf{e}_j)}{\sum_{i=1}^{n} exp(\mathbf{e}_j)}, \tag{22}$$

$$\mathbf{m} = \sum_{i=1}^{n} \alpha_j \mathbf{W}_{o^*}\mathbf{o}_j^*, \tag{23}$$

where $\mathbf{W}$ is the weight matrix, and α_j is the weight of the j-th position of the document. $\mathbf{m}$ is the attention vector, that is, the output of the attention mechanism.

Finally, TextGRNN feed the attention vector into the softmax layer to predict. The loss ℓ is minimized through the cross-entropy function:

$$\widehat{y} = \mathrm{softmax}(\mathbf{W}_m\mathbf{m} + \mathbf{b}_m), \tag{24}$$

$$\ell = -\sum_i y_i \log(\widehat{y}_i), \tag{25}$$

where $\mathbf{W}$ and $\mathbf{b}$ are weights and bias, and y_i is the i-th element of the one-hot label.

4 Experiments

4.1 Datasets

We adopt three datasets: R8, R52[1] and MR[2]. R8 and R52 are two subsets of Reuters 21578 datasets that have 8 and 52 categories, respectively. MR is a collection of polarized movie reviews and has 2 categories. The statics of these datasets are demonstrated in Table 1.

4.2 Baselines

To investigate the advance of the model proposed in this paper, we selected 6 existing models as the baselines for comparison. Some models have resulted directly from (Zhang et al., 2020)

- CNN: Kim (2014) performs convolution and max-pooling operation on word embeddings to get a representation.
- LSTM: Liu et al. (2016) use the last hidden state as the representation of the text. Bi-LSTM is a bi-directional LSTM.
- fastText: Joulin et al. (2017) average word or n-gram embeddings as document embeddings.
- TextGCN: Yao et al. (2019) build a large graph for the whole corpus to text classification.
- Text Level Graph: Huang et al. (2019) and TextING (Zhang et al., 2020) both build corresponding graphs for every document to text classification.

[1]http://disi.unitn.it/moschitti/corpora.htm

[2]http://www.cs.cornell.edu/people/pabo/movie-review-data/

4.3 Experiment Set-up

We have obtained the training set and test set. The actual training set and validation set are obtained by splitting the training set into the ratio 9:1. We adjust the hyperparameters according to the performance of the validation set, and adjust the learning rate to 0.01 via the Adam optimizer (Kingma and Ba, 2015). Dropout is set as 0.5, and the number of layers for Bi-LSTM is 2. Training will be performed in 300 epochs. The default size of the sliding window is 7.

The word embeddings are initialized with Glove[3] (Pennington et al., 2014) with 300 dimensions, and The out-of-vocabulary(OOV) words' embedding were randomly selected from a uniform distribution [-0.01, 0.01].

4.4 Result

Table 2 shows the consequence of TextGRNN against other baseline methods, and our model is state-of-the-art.

We observe that the GNN-based approaches perform better than other models, suggesting that some closely related but distant words can interact through the graph. Their interaction frequencies are reflected by the edge weights.

We also note that TextGCN, Huang et al. (2019) and TextING can not represent the word order of the document. In order to learn the semantic information of the document, our proposed TextGRNN model upgrades the embedding of word nodes through the graph, then transmits them to the Bi-LSTM layer as well as attention layer according to the word order. Therefore, our model performs better.

4.5 Attention Visualisation

To understand how TextGRNN learns the semantic information of the document, we visualize the attention layer (Yang and Zhang, 2018) on MR as shown in Table 3. Our model pays more attention to words that contribute positively to sentiment analysis and ignores noise words that

[3]http://nlp.stanford.edu/data/glove.6B.zip

Table 1: The statistics of the datasets including MR, R8 and R52. The vocab means the number of unique words in a document.

Dataset	Docs	Training	Test	class	Avg. Vocab	Avg. Length
MR	10662	7108	3554	2	18.46	6
R8	7674	5485	2189	8	41.25	65.72
R52	9100	6532	2568	52	44.02	69.82

Table 2: Test accuracy of various models on three datasets. Our model's mean ± standard deviation is based on 10 time runs reported.

Model	R8	R52	MR
CNN(Non-static)	95.71 ± 0.52	87.59 ± 0.48	77.75 ± 0.72
Bi-LSTM	96.31 ± 0.33	90.54 ± 0.91	77.68 ± 0.86
fastText	96.13 ± 0.21	92.82 ± 0.09	75.14 ± 0.20
TextGCN	97.07 ± 0.10	93.56 ± 0.18	76.74 ± 0.20
Huang et al. (2019)	97.80 ± 0.20	94.60 ± 0.30	-
TextING	98.04 ± 0.25	95.48 ± 0.19	79.82 ± 0.20
TextGRNN	98.42 ± 0.08	95.99 ± 0.10	80.82 ± 0.37

negatively contribute to sentiment analysis. For example, TextGRNN highlights `'interesting'` and misses `'bad'` in the positive review. In contrast, TextGRNN emphasizes `'little'` and ignores `'explicit'` in the negative review.

Table 3: Attention visualization of positive and negative reviews in MR.

Labels	Reviews
Positive	it is usually a bad sign when directors abandon their scripts and go where the moment takes them , but olympia , wash , based filmmakers anne de marcken and marilyn freeman did just that and it is what makes their project so interesting
Negative	aside from showing us in explicit detail how difficult it is to win over the two drink minimum crowd , there is little to be learned from watching 'comedian'

5 Ablation Study

5.1 Model Variant

There are four ways to change the structure of the TextGRNN, called Model-V1, Model-V2, Model-V3 and Model-V4. Table 4 shows the result of these model variants, TextING, and our model.

Model-V1 Instead of feeding the updated word nodes into Bi-LSTM, Model-V1 aggregates the representations of all word nodes to generate predictions. Like TextING, the readout function are:

$$\mathbf{v}_t^{\nu} = \sigma(f_1(\mathbf{v}_t^{\nu})) \odot \tanh(f_2(\mathbf{v}_t^{\nu})), \tag{26}$$

$$\mathbf{v}_{\mathcal{G}} = \frac{1}{|\mathcal{V}|}\sum_{\nu\in\mathcal{V}} \mathbf{v}_t^{\nu} + \text{Maxpooling}\left(\mathbf{v}_t^{1} \ldots \mathbf{v}_t^{\mathcal{V}}\right), \tag{27}$$

where f_1 and f_2 are two multilayer perceptrons (MLP). Maxpooling is a max-pooling function.

Model-V2 Model-V2 changes TextGRNN with the different ways of building graphs, where Word Co-occurrence replaces Visibility Graph.

Model-V3 To exam whether the attention layer can work, Model-V3 is designed without the attention layer based on our models.

Model-V4 We build the graph for the document by Visibility Graph, in which the edges between words are determined by counting the word length. The word length is a natural attribute and is replaced by the TF-IDF (Wu et al., 2008), an artificial feature of words, in model-V4.

5.2 Result

Table 4 exhibits that TextGRNN, Model-V1, Model-V3 and Model-V4 can work better than

Table 4: Test accuracy of various models on three datasets. All models' mean ± standard deviation is based on 5 time runs reported.

Model	R8	R52	MR
TextING	98.01 ± 0.11	95.41 ± 0.17	79.87 ± 0.30
Model-V1	98.04 ± 0.06	95.51 ± 0.15	79.91 ± 0.15
Model-V2	98.17 ± 0.09	95.81 ± 0.16	80.04 ± 0.18
Model-V3	98.33 ± 0.15	95.80 ± 0.11	80.71 ± 0.32
Model-V4	98.27 ± 0.09	95.82 ± 0.07	80.10 ± 0.11
TextGRNN	98.43 ± 0.11	96.02 ± 0.11	80.84 ±0.39

Model-V1 and TextING, which reveals that models with the Bi-LSTM layer can learn the semantic information of the document about word order. The accuracy of Model-V1 is higher than TextING on MR, R8, and R52, suggesting that compared with Word Co-occurrence, Visibility Graph can work very well. The outcome of Model-V3 indicates that if our model discards the attention mechanism, then essential words in the document will not be highlighted, and the model's performance will decrease. Comparing with Model-V4 and TextGRNN, the result shows that artificial features are not necessarily preferable to the natural properties of words.

5.3 Graph Density

Figure 2 exhibits the graph density of Visibility Graph and Word Co-occurrence for different window sizes on MR, R8 and R52. Figure 3 shows the performance of TextGRNN and Model-V2 with a varying window size on MR, R8 and R52. The result proves that when window size increases, the rapid growth of graph density of Word Co-occurrence results in the graph nodes being over-smooth. Thus, the semantic information of the document is destroyed, and the performance of Model-V2 is lower than TextGRNN. However, Visibility Graph constrains the graph density to a lower level, where nodes achieve the information from more neighbors and avoid destroying the semantic information. Hence, the accuracy of TextGRNN still improves when the window size surpasses 3.

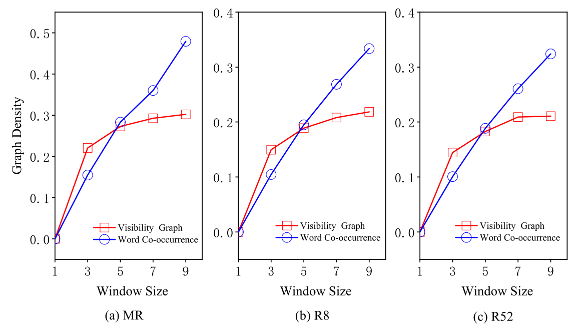

Figure 2: Graph density with varying window size

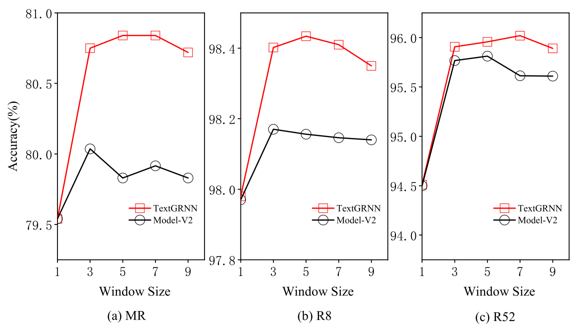

Figure 3: Accuracy with varying window size

6 Conclusion

Unlike the existing GNN-based methods that only focus on the interaction between words in the text, we propose a model for text classification, which can be conscious of the word order based on retaining the advantages of the GNN-based models. In addition, we construct the graph for every document through Visibility Graph, which is helpful to decrease the graph density and prevent graph nodes from becoming over-smooth. Experiments prove that our model has an excellent ability to express semantic information.

However, in this work, we only discuss two cases where word length or word TF-IDF is the data point of time series without exploring other ways to calculate data points. In the future, finding more methods to map the document into time series will be the focus of our work.

References

M. Ausloos. 2012. Generalized hurst exponent and multifractal function of original and translated texts mapped into frequency and length time series. *Phys. Rev. E*, 86:031108.

Peter F. Brown, Vincent J. Della Pietra, Peter V. de Souza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Comput. Linguistics*, 18(4):467–479.

Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. 2020. Be more with less: Hypergraph attention networks for inductive text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4927–4936. Association for Computational Linguistics.

Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE.

Lianzhe Huang, Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2019. Text level graph neural network for text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3442–3448. Association for Computational Linguistics.

Ignacio Iacobacci and Roberto Navigli. 2019. Lstmembed: Learning word and sense representations from a large semantically annotated corpus with long short-term memories. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1685–1695. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomás Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 427–431. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2267–2273. AAAI Press.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016. Gated graph sequence neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2873–2879. IJCAI/AAAI Press.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020. Tensor graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8409–8416.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari.

Giannis Nikolentzos, Antoine J.-P. Tixier, and Michalis Vazirgiannis. 2020. Message passing attention networks for document understanding. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8544–8551. AAAI Press.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

François Rousseau, Emmanouil Kiagias, and Michalis Vazirgiannis. 2015. Text categorization as a graph classification problem. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1702–1712. The Association for Computer Linguistics.

Mutua Stephen, Changgui Gu, and Huijie Yang. 2015. Visibility graph based time series analysis. *PLOS ONE*, 10:1–19.

Denner S. Vieira, Sergio Picoli, and Renio S. Mendes. 2018. Robustness of sentence length measures in written texts. *Physica A: Statistical Mechanics and its Applications*, 506:749–754.

Zhengjue Wang, Chaojie Wang, Hao Zhang, Zhibin Duan, Mingyuan Zhou, and Bo Chen. 2020. Learning dynamic hierarchical topic graph with graph convolutional network for document classification. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3959–3969. PMLR.

Ho Chung Wu, Robert Wing Pong Luk, Kam-Fai Wong, and Kui-Lam Kwok. 2008. Interpreting TF-IDF term weights as making relevance decisions. *ACM Trans. Inf. Syst.*, 26(3):13:1–13:37.

Man Wu, Shirui Pan, Xingquan Zhu, Chuan Zhou, and Lei Pan. 2019. Domain-adversarial graph neural networks for text classification. In *2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, pages 648–657. IEEE.

Qianqian Xie, Jimin Huang, Pan Du, Min Peng, and Jian-Yun Nie. 2021. Inductive topic variational graph auto-encoder for text classification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4218–4227. Association for Computational Linguistics.

Jie Yang and Yue Zhang. 2018. NCRF++: an open-source neural sequence labeling toolkit. In *Proceedings of ACL 2018, Melbourne, Australia, July 15-20, 2018, System Demonstrations*, pages 74–79. Association for Computational Linguistics.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7370–7377. AAAI Press.

Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. 2020. Every document owns its structure: Inductive text classification via graph neural networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 334–339. Association for Computational Linguistics.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics.