DEEP ATTENTION POOLING GRAPH NEURAL NETWORKS FOR TEXT CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph Neural Networks (GNN) is a classical method that has been applied to document classification as a compelling message-passing framework inside and between documents. Consider the graph-based models are transductive when representing the documents as nodes in one graph(inter-documents), and require high memory and time efficiency to employ the GNN to each document after aligning the documents to the longest one(intra-documents). This paper proposes a novel method named Deep Attention Pooling Graph Neural Networks (DAPG) to use the structure of each document for inductive document classification. The attention pooling layer (APL) in DAPG adaptively selects nodes to form smaller graphs based on their scalar attention values to alleviate resource consumption. Additionally, regarding the structural variation, a fresh dual adjacency matrix for individual graphs based on the word co-occurrence and the word distance has been built to conquer the sparsity and keep stability after pooling. Experiments conducted on five standard text classification datasets show that our method is competitive with the state-of-the-art. Ablation studies reveal further insights into the impact of the different components on performance.

1 INTRODUCTION

GNN has demonstrated great capability for various challenging NLP tasks. As in the text classification field, where words in the document have locality and order information but with ambiguities and the unstructured relationship between documents which hinders the traditional CNN Technicolor et al. (2017), Haykin & Kosko (2001) and RNN Mikolov et al. (2010) but corresponds to the instincts of graph convolutional operations. Defferrard et al. (2016) first employed Graph Convolutional Neural Networks (GCN) in the text classification task. Further, Yao et al. (2018) improved the work by employing Graph Convolutional Networks Kipf & Welling (2016) on article nodes and word nodes in one graph, which turns the text classification problem into article nodes classification. Moreover, Huang et al. (2019) introduced a message-passing mechanism to improve the TextGCN and gain impressive results. Zhang et al. (2020) inherits the Gate Graph Neural Networks in Nikolentzos et al. (2020) as an information aggregator which utilizes the context inside each document.

The core idea of the message-passing framework is recursive information aggregation of neighborhoods. The concept of message passing over graphs has been around for many years. And most of the spectral GNN are based on it. Notable examples include Kipf & Welling (2016), Gilmer et al. (2017), and Xu et al. (2018) which are applied to text classification, bio-information, and social network data with great success. Meanwhile, the spatial GNN methods (Hamilton et al. (2017) ,Chen et al. (2018),Velikovi et al. (2017)) based on message aggregation applied to text classification are also

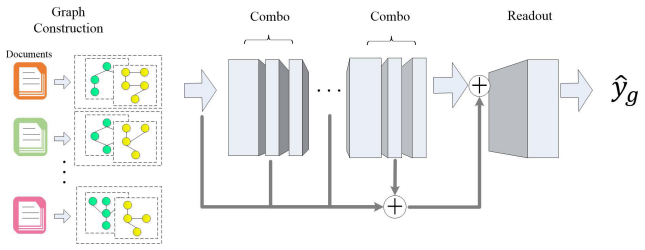

Figure 1: An illustration of the DAPG network configuration.

competitive. However, most of the work focused on message passing between the documents and neglected the context-aware word relations within each document. They are inherently transductive and have difficulty with inductive learning. Nikolentzos et al. (2020), Zhang et al. (2020) develop an expressive intra-document message-passing GNN, which is tailored to document understanding with the word co-occurrence networks of each document and obtain the compressive result in text classification.

However, there are two drawbacks of the intra-document GNN models. First, it needs to align the documents according to the longest one to build the adjacency matrices which causes high memory consumption and inefficiency. Second, after the padding, the only co-occurrence adjacent matrix is often very sparse, which causes the information propagation difficult and slow convergence. Therefore, We propose a novel text classification method based on GNN with attention pooling to release the memory consumption and dual adjacency matrix which merges two individual adjacency matrices to conquer the sparsity. To sum up, the main contributions of this paper are as follows:

• A novel deep GNN framework with Attention Pooling is proposed to alleviate memory consumption and boost efficiency and accuracy.

• We propose a new graph constructor which combines the word co-occurrence and the word distance to build a dual adjacency matrix. The new adjacency matrix conquers the sparsity and keeps the model performance stable, even though the graph structure varied after pooling.

• The experimental results on text classification demonstrate the effectiveness of our proposed method as compared to previous methods. Three databases conquer the state-of-the-art and two equal of them.

2 METHOD

An overview of the DAPG architecture is presented in Fig. 1. The model comprises graph construction, multiple consecutive combos, and a readout layer. The graph construction builds the graphs for each document, then feeds them to the combos. Several cascade combos are applied to these graphs and the outputs are concatenated for the latter layer. Finally, the readout layer generates the embeddings of the documents for classification. In this section, we describe the details of the main components of the model.

2.1 GRAPH CONSTRUCTION

To represent the document as graph-structured data, we build a graph with two types of edges which denoted as $G = (V, E)$, V represents the nodes and each node represents an unique word in the document. $E = f(E_1, E_2)$ represents the edges between the nodes, and E_1, E_2 represent two edge

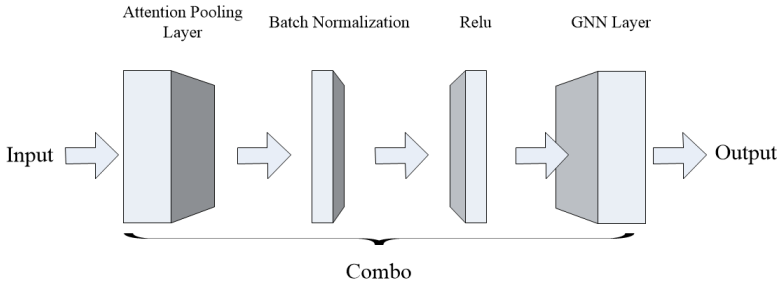

Figure 2: The structure of the combo

types. E_1 is the words co-occurrence in the statistical word co-occurrence network (Mihalcea & Tarau (2004)) with a sliding window overspanning sentences. The bigger size of the sliding window made E_1 dense, but vague the structure of the document. On the contrary, the small size made it sparse which impedes information propagation. Here we set it as 3 as default. E_2 is calculated from the Generalized Mahalanobis distance between words as:

$$D(x_i, x_j) = \sqrt{(x_i - x_j)^T M (x_i - x_j)}, \tag{1}$$

$D(x_i, x_j)$ represents the generalized Mahalanobis distance between x_i and x_j where x_i and x_j are two neighboring words. Where M is trainable weights and set to be I in this paper to improve the efficiency which degenerating D into Euclid distance. Then the distance is applied to calculate the Gaussian kernel:

$$G(x_i, x_j) = \exp(-D(x_i, x_j)/(2\sigma^2)), \tag{2}$$

where the normalized G is the adjacency matrix of E_2 edges. We add it to adjacent matrix A_1 of E_1 edges to generate A as the final graph adjacency matrix. The formula is as follows:

$$A = \text{norm}(\text{add}(A_1 + \alpha * G)). \tag{3}$$

Where α is a trainable coefficient. We set α as 1 and define A as a dual adjacency matrix for computing efficiency. And define A as an incremental-dual adjacency matrix when $0 \leq \alpha \leq 1$ where G is as an incremental of A_1.

2.2 DAPG COMBO

Several cascade combos with the same structure are stacked in DAPG. As illustrated in Fig. 1, the combo contains an attention pooling layer (APL), followed by a GNN layer, with a BN and Relu non-linearity in between. The input is a graph processed by the prior graph constructor or combo. The attention pooling layer is applied to the input graph to select nodes to retain. Then the GNN layer is employed to update the nodes from their neighbors as an aggregator.

2.2.1 ATTENTION POOLING LAYER

The APL plays an important part in the DAPG of down-sampling the nodes. It adaptively selects a subset of nodes according to the scalar attention score Gao & Ji (2022). The layer employs a trainable projection method to project all node features to one dimension. Then perform max pooling for node selection, and α ($\alpha \leqq 1$) percent of the largest scalar projection values are selected to retain for the

new graph. Also, the adjacency matrix will be rebuilt based on the retained nodes. The formulas of the node selection are as follows:

$$\begin{aligned}
\mathbf{y} &= \text{norm}(\text{Proj}(\mathbf{X}^{\ell})), \\
\mathbf{idx} &= \text{top}_k(\mathbf{y}, \alpha), \\
\tilde{\mathbf{y}} &= \text{sigmoid}(\mathbf{y}(\mathbf{idx})), \\
\tilde{\mathbf{X}}^{\ell} &= \mathbf{X}^{\ell}(\mathbf{idx}, :), \\
\mathbf{A}^{\ell+1} &= \mathbf{A}^{\ell}(\mathbf{idx}, \mathbf{idx}), \\
\mathbf{X}^{\ell+1} &= \tilde{\mathbf{X}}^{\ell} \odot (\tilde{\mathbf{y}} \mathbf{1}_C^{\mathrm{T}}).
\end{aligned} \tag{4}$$

After aligning all the graphs, suppose there are N nodes in one graph and each node contains C features. Where $\mathbf{A}^{\ell} \in \mathbb{R}^{N \times N}$ and $\mathbf{X}^{\ell} \in \mathbb{R}^{N \times C}$ are the adjacency matrix and feature matrix of the current layer. $\mathbf{y}$ is the normalized projection of $\mathbf{X}^{\ell}$. The $\text{top}_k()$ select k largest nodes and return the index $\mathbf{idx}$ according to the α rate. $\tilde{\mathbf{X}}^{\ell}$ and $\mathbf{A}^{\ell}$ are the new nodes and new adjacent matrix which are downsampled from $\mathbf{X}^{\ell}$ and $\mathbf{A}^{\ell}$ based on the $\mathbf{idx}$. The nonlinearity sigmoid function is applied to generate attention scores $\tilde{\mathbf{y}}$. Finally, the element-wise multiplication is applied on the $\tilde{\mathbf{X}}^{\ell}$ with $\tilde{\mathbf{y}}$, then output the new embedding $\mathbf{X^{l+1}}$.

2.2.2 GNN LAYER

After the graphs were compressed and updated by APL, the GNN layer is employed as a message-passing framework between the nodes in the graphs. In this paper, We compare the Classic GNN (Kipf & Welling (2016)) with the Gated Graph Neural Networks (Li et al. (2015)) as a GNN layer, getting similar accuracy but more efficiency. The concrete details will be elaborated on in the ablation part. The formula of the GNN aggregation is as follows:

$$\mathbf{X}^{l+1} = f(\tilde{\mathbf{A}}^{\ell}, \mathbf{W}^{\ell}, \mathbf{X}^{\ell}), \tag{5}$$

where $\tilde{\mathbf{A}}^{\ell} \in \mathbb{R}^{N \times N}$ is the normalized adjacency matrix. We pre-process the $\tilde{\mathbf{A}}^{\ell}$ as $\tilde{\mathbf{D}}^{\ell-\frac{1}{2}} \mathbf{A}^{\ell} \tilde{\mathbf{D}}^{\ell-\frac{1}{2}}$ with $\mathbf{A}^{\ell} = I + \mathbf{A}^{\ell}$. $\mathbf{A}^{\ell} \in \mathbb{R}^{N \times N}$ is the adjacency matrix, $\tilde{\mathbf{D}}^{\ell}$ is the degree matrix of $\tilde{\mathbf{A}}^{\ell}$ as $\tilde{\mathbf{D}}_{ii}^{\ell} = \sum_j \tilde{\mathbf{A}}_{ij}^{\ell}$ and $\mathbf{X}^{\ell} \in \mathbb{R}^{N \times D}$ is the nodes feature matrix of the graph from former layer. $\mathbf{W}^{\ell} \in \mathbb{R}^{D \times D}$ is a transformation matrix to be trained. f is a non-linear function, in this paper, it set as Relu as default.

2.3 READOUT LAYER

The readout layer merges the averaging and the max-pooling of the nodes in graphs to generate the graph-level representation of the documents. The formulas are defined as follows:

$$\begin{aligned}
\mathbf{X} &= \text{concat}(\mathbf{X}^1, \mathbf{X}^2 \ldots \mathbf{X}^l), \\
\mathbf{X}_{att} &= \sigma(\text{softmax}(\mathbf{X}\mathbf{W}_a) \odot \tanh(\mathbf{X}\mathbf{W}_b)), \\
\mathbf{Y}_g &= \frac{1}{|n|} \sum_{n=1}^{\infty} \mathbf{X}_{att} + \text{maxpooling}(\mathbf{X}_{att}),
\end{aligned} \tag{6}$$

where $\mathbf{X}$ is the concatenation of $\{\mathbf{X}^1, \mathbf{X}^2 \dots \mathbf{X}^l\}$ which are the nodes feature matrices output from cascade combos. The $\mathbf{X}_{att}$ is the output of a sub-layer that works as a global self-attention mechanism (Lin et al. (2017)). $\mathbf{W}_a \in \mathbb{R}^{N \times D}$ and $\mathbf{W}_b \in \mathbb{R}^{N \times D}$ are the trainable weight matrices. $softmax$ performs as soft attention to generate an alignment vector to dot product with the latter part that has been non-linear transformed via $tanh$. σ is the no-linear activation function. Then the max-pooling and the average of $\mathbf{X}_{att}$ are concatenated to generate the final representation of the graph.

Regarding the text classification task in this paper, we add a fully connected layer with softmax to generate the class probability of the documents and choose Cross-entropy loss to train the model:

$$\begin{aligned} \mathbf{Y}_l &= \text{softmax}(\mathbf{W}\mathbf{Y}_\mathbf{g} + \mathbf{b}), \\ \mathcal{L} &= -\sum_i \mathbf{Y}_{li} \log(\mathbf{Y}_{li}) \ . \end{aligned} \tag{7}$$

Where W and b are trainable weights and bias, $\mathcal{L}$ is the model loss.

3 EXPERIMENT

In this section, to evaluate the overall performance of DGAP, we compare our model with the previous state-of-the-art models on the document classification task. Experimental results show that our method achieved competitive results. Some ablation studies are performed to examine the contributions of APL and the dual adjacency matrix on performance improvements. We conduct studies on the relationship between network depth and node classification performance and illustrate the dynamics of the adjacency matrix after pooling.

3.1 DATASETS

In this paper, five datasets Yao et al. (2018) have been adopted to evaluate our model in document classification for consistency. MR: binary classification of the movie sentiment as positive and negative. Ohsumed: Multi-class classification for 23 cardiovascular disease categories of medical abstracts. R8 and R52: Multi-class classification for Reuters Newswire articles. R8 is 8 categories and R52 is 52 categories. 20NG: no overlap news assembling which contains 18,846 documents evenly categorized into 20 categories.

3.2 BASELINES

To evaluate our model, we choose six models as baselines:

TextCNN Kim (2014) is a classic graph-based text classification model that aggregates the information of nodes through GCN. TextGCN Yao et al. (2018) is the traditional deep learning method that employs GCN on hybrid nodes graph in text classification. FastText Joulin et al. (2017) selects the word layer by layer to build graphs via Mentocalo downsampling. SWEM Shen et al. (2018) is a Simple Word-Embedding-based Model which consists of parameter-free pooling operations. TextING Zhang et al. (2020) uses the structure of each document and passes messages between words via gate GCN.

Model	MR	R8	R52	Ohsumed	20NG
TextCNN(non-static)	77.75 ± 0.72	95.71 ± 0.52	87.59 ± 0.48	58.44 ± 1.06	82.15 ± 0.52
FastText	75.14 ± 0.20	96.13 ± 0.21	92.81 ± 0.09	57.70± 0.49	79.67 ± 0.29
TextRNN	77.68 ± 0.86	96.31 ± 0.33	90.54 ± 0.91	49.27 ± 1.07	75.43 ± 1.72
SWEM	76.65 ± 0.63	95.32 ± 0.26	92.94 ± 0.24	63.12 ± 0.55	85.16 ± 0.29
TextGCN	76.74 ± 0.20	97.07 ± 0.10	93.56 ± 0.18	68.36 ± 0.56	86.34 ± 0.06
TextING	79.82 ± 0.20	98.04 ± 0.25	95.48 ± 0.19	70.42 ± 0.39	85.47 ± 0.56
DAPG	**80.22 ± 0.31**	**98.21 ± 0.27**	**95.97 ± 0.35**	**71.47 ± 0.51**	**87.16 ± 0.19**

Table 1: Document Classification accuracies(%).The mean and standard deviation in the five datasets. Best performance per column in bold. Note that some baseline results are from (Yao et al. (2018)).

Model	MR	20NG
Co-occurrence	79.75	86.38
Word distance	78.61	86.02
Dual	80.22	87.16
Incremental-Dual	80.32	87.23

Table 2: Document Classification accuracies(%) of Co-occurrence adjacency matrix, word distance adjacency matrix, Dual adjacency matrix and Incremental-Dual adjacency matrix.

3.3 Experimental Set-up

For fairness and consistency, we randomly split the training set of the datasets into a ratio of 9:1 for training and validation, set the rates of all the pooling layers as 0.6, and stack two combos as default for efficiency. Other parameters are tuned according to the performance of the validation set. We use the Adam optimizer with a learning rate of 0.005 and the dropout rate of 0.5, choose the pre-trained GloVe(Pennington et al. (2014)) with 300 dimensions as the word embedding, and the out-of-vocabulary (OOV) words are randomly initialized from a uniform distribution [-0.01, 0.01].

3.4 Performance study

We compare our DAPG to other models in terms of text classification accuracy. Table 1 summarizes the results on datasets MR, Ohsumed, R8, R52, and 20NG. For baseline values listed, they are state-of-the-art on these datasets. We observe that the DAPG achieves consistently better performance than other models. When compared to TextING directly, the DAPG significantly improves performance on three datasets by margins of 0.5 (R52), 1.7 (20NG), and 1.05 (Ohsumed), respectively, and performs slight enhancement on the other two datasets. Due to the short documents of the MR dataset (average length is 18 and 44 after the padding to align), APL selects almost all the words to retain and leave the only dual adjacency matrix enhanced the graph density, DAPG gets a tiny improvement. Since R8 and R52 are simple, all the GNNs baselines perform satisfactorily. Ohsumed is a dataset of long text with a small training set that restricts the generalization ability of the models except for the GNNs. Meanwhile, most models perform reasonably on the long document dataset 20NG with a big training set except TextRNN is in line with that DAPG is superior to TextING which is based on GGNN(RNN architecture GNN). These results demonstrate the effectiveness of our model.

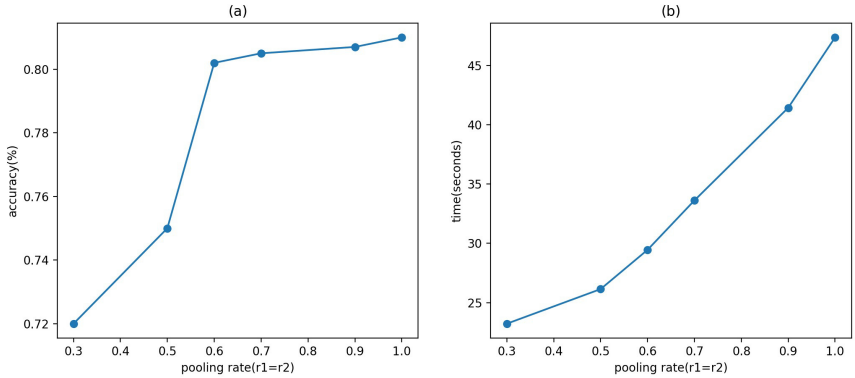

Figure 3: (a) Test accuracy by varying the pooling rates of the MR dataset. (b) Time of one epoch on MR dataset by varying the combos pooling rates

3.5 Ablation studies

Dual Adjacency matrix. Table 2 illustrates that the dual adjacency matrix improves the performance of DAPG. In the MR dataset, the padding adjacency matrics of word co-occurrence of the too-short documents are sparse. After pooling, the trivial variation of the structure causes the approximate performance between word co-occurrence and dual adjacency matrix. The only word distance adjacency matrix vague the structure of the documents and even perform worse than the co-occurrence adjacency matrix. For the long text dataset 20NG, the equal performance of the two single adjacency matrices and the evident improvements of the dual adjacency matrix indicates it captured the non-overlapping information of the other two adjacency matrices. The last row shows the trainable coefficient of the Incremental-Dual adjacency matrix increases the performance slightly but causes more memory consumption and convergence slowness.

Attention Pooling rate. The pooling rate decides the proportion of nodes in a graph to retain. To quantify the relationship between the pooling rate and accuracy, we set two combos as default in DAPG and the two rates($r1$ and $r2$) of the APL in the combos as the same. The cascade combos output the retained nodes as $N(r1 \times r2)$ indicates the squeeze of the focus. As shown in Fig. 3(a), the tendency of the accuracy curve grows fast until 0.6×0.6 and becomes flat in the rear. It indicates the mild relation of discard nodes filtrated by APL with document comprehension. When the rates are 1×1, the combo degrades to an attention layer. Fig. 3(b) demonstrates the variation of one epoch time as an approximate square curve due to the matrix multiplication in the combos. The variation is flat in the beginning but steep in the rear. 0.6×0.6 are appropriate rates for both accuracy and efficiency.

Combo depth. As described in (Kipf & Welling (2016)), two layers are appropriate for GCN. We varied the combo depth from 1 to 4 and set the pooling rate as 0.6 as the default for all APL. Fig. 4 shows that two and three combos get approximate accuracy but the time increased pari passu the combo depth indicates two combos outperform all combo variants. It explained that the excessive GCN layer induced over-smoothing and overemphasizing of the pooling caused the contextual missing.

Effictive of GNNs. To verify the effectiveness of the GNNs, we compare the classical GNN(Kipf & Welling (2016)), Chebyshev(2 layers) (A et al. (2011)), GGNN(Li et al. (2015)), and the Dense layer(fully connected neural network) as a message-passing layer in DAPG. As shown in Table 3, the GNNs outperform the Dense layer(even increase the trainable weights) on both MR and Ohsumed

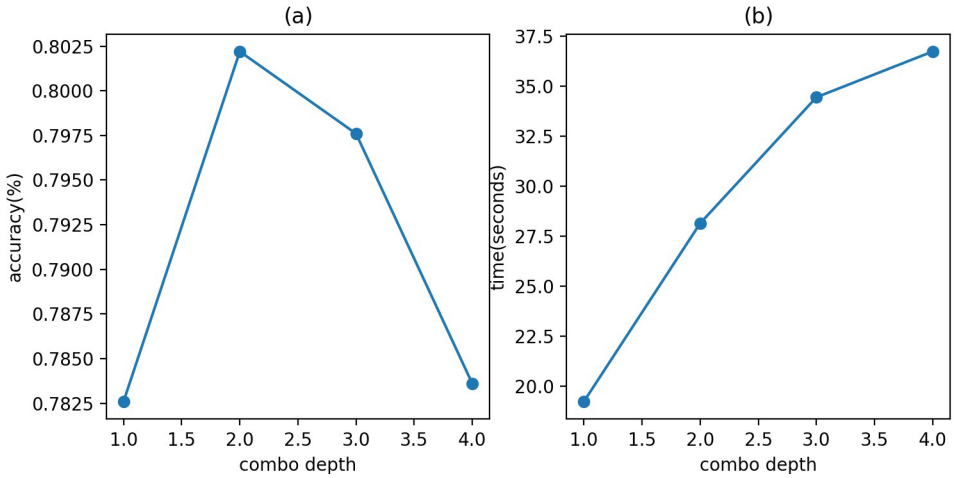

Figure 4: (a) Accuracy with the combo depth variation of MR dataset. (b) One epoch time of the combo depth variation of MR dataset.

Dataset	**MR**		**Ohsumed**	
	Accuracy	Time	Accuracy	Time
GNN	80.12 ± 0.20	27.5 + 5	71.04 ± 0.36	42.6 + 3
Chebyshev	80.22 ± 0.21	28.7 + 5	71.47 ± 0.51	45.7 + 4
GGNN	80.35 ± 0.42	47.2 + 7	70.88 ± 0.33	80.2 + 11
Dense	76.05 ± 0.30	26.2 + 4	67.08 ± 0.71	35.7 + 7

Table 3: Document Classification accuracies(%) and the one epoch time(s)(+ means the disturbance of the shortest epoch) of GNN ,Chebyshev, GGNN, Dense.

datasets in accuracy, suggesting the validity of the graph methods. Additionally, the performance of the classical GCN is similar to GGNN (the margin is about 0.1 on MR and 0.4 on Ohsumed) but more efficient (depending on the length of the documents, about 20 seconds faster in one epoch on MR and 35 seconds on Ohsumed)

Case study. Fig. 5 illustrates the visual attention pooling layer. The count of red highlighted words in Fig. 5(a) is proportional to the pooling rate of the first APL. The blue highlighted words in Fig. 5(b) of the second APL are words retained from the first APL concerning the cascade connection. Intuitively, The blue words are more positively correlated to the document label than the red words. Concatenating the filtrated words advanced the performance significantly in sentiment analysis which demonstrates the effectiveness of the APLs.

what makes the movie work to an admittedly limited extent is the commitment of two genuinely engaging performers weaver and lapaglia are both excellent , in the kind of low key way that allows us to forget that they are actually movie folk

(a)

what makes movie work admittedly limited extent commitment two genuinely engaging performers weaver lapagli excellent kind low key way allows us forget they actually movie folk

(b)

Figure 5: The length of the padding documents of MR dataset is 44 for alignment, and the pooling rate is 0.6 × 0.6 as default. 26 words are reserved for the first APL, and 16 words are retained for the second APL. (a) Filtrated words of the first APL; (b) Filtrated words of the second APL.

4 CONCLUSION

This paper presents DAPG, a deep GNN model with APL(attention pooling layers) that leverages the structure of each document for inductive test classification. DAPG achieves state-of-the-art performance on five datasets which significantly surpasses previous best methods. As a primary element of DAPG, the document structure has shown to be effective and efficient in text classification. Therefore we look forward to investigating its use in other natural language processing tasks.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grants 62106020; the China Postdoctoral Science Foundation under Grant 2021M690355.

REFERENCES

David K. Hammond A, Pierre Vandergheynst B, and Rémi Gribonval c. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):121–150, 2011.

J. Chen, T. Ma, and C. Xiao. Fastgcn: Fast learning with graph convolutional networks via importance sampling. In *International Conference on Learning Representations*, 2018.

Michal Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Neural Information Processing Systems*, 2016.

H. Gao and S. Ji. Graph u-nets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (9):4948–4960, 2022.

J. Gilmer, Samuel S Schoenholz, Patrick F Riley, O. Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, 2017.

W. L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Neural Information Processing Systems*, 2017.

Simon Haykin and Bart Kosko. *GradientBased Learning Applied to Document Recognition*. Wiley-IEEE Press, 2001.

L. Huang, D. Ma, S. Li, X. Zhang, and H. Wang. Text level graph neural network for text classification. In *Empirical Methods in Natural Language Processing*, 2019.

A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. In *Association for Computational Linguistics*, 2017.

Y. Kim. Convolutional neural networks for sentence classification. 2014.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2016.

Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel. Gated graph sequence neural networks. 2015.

Z. Lin, M. Feng, Cnd Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. 2017.

R. Mihalcea and P. Tarau. Textrank: Bringing order into text. 2004.

T. Mikolov, M Karafiát, L. Burget, J Cernocký, and S. Khudanpur. Recurrent neural network based language model. *ACM*, pp. 896–899, 2010.

G. Nikolentzos, A. Tixier, and M. Vazirgiannis. Message passing attention networks for document understanding. In *The Association for the Advancement of Artificial Intelligence*, 2020.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*, 2014.

D. Shen, G. Wang, W. Wang, Martin Renqiang, and L. Carin. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. 2018.

Ts/ Technicolor, Sor Related, Ts/ Technicolor, and Sor Related. Imagenet classification with deep convolutional neural networks [50]. *Communications of the ACM*, 60(6):84–90, 2017.

Petar Velikovi, G. Cucurull, A. Casanova, A. Romero, P Liò, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2017.

K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? 2018.

L. Yao, C. Mao, and Y. Luo. Graph convolutional networks for text classification. In *The Association for the Advancement of Artificial Intelligence*, 2018.

Y. Zhang, X. Yu, Z. Cui, S. Wu, Z. Wen, and L. Wang. Every document owns its structure: Inductive text classification via graph neural networks. In *Association for Computational Linguistics*, 2020.