Deep Attention Diffusion Graph Neural Networks for Text Classification

Yonghao Liu[†], **Renchu Guan**[†], **Fausto Giunchiglia**[‡], **Yanchun Liang**[◇], **Xiaoyue Feng**[†*]

[†]Key Laboratory of Symbolic Computation and Knowledge Engineering of the Ministry of Education, College of Computer Science and Technology, Jilin University, China

[◇]Zhuhai Sub Laboratory, Key Laboratory of Symbolic Computation and Knowledge Engineering of the Ministry of Education, Zhuhai College of Science and Technology, China

[‡]DISI, University of Trento, Italy

yonghao20@mails.jlu.edu.cn {guanrenchu, ycliang, fengxy}@jlu.edu.cn

fausto.giunchiglia@unitn.it

Abstract

Text classification is a fundamental task with broad applications in natural language processing. Recently, graph neural networks (GNNs) have attracted much attention due to their powerful representation ability. However, most existing methods for text classification based on GNNs consider only one-hop neighborhoods and low-frequency information within texts, which cannot fully utilize the rich context information of documents. Moreover, these models suffer from over-smoothing issues if many graph layers are stacked. In this paper, a **D**eep **A**ttention **D**iffusion **G**raph **N**eural **N**etwork (DADGNN) model is proposed to learn text representations, bridging the chasm of interaction difficulties between a word and its distant neighbors. Experimental results on various standard benchmark datasets demonstrate the superior performance of the present approach.

1 Introduction

Text classification, as an important and fundamental task in the field of natural language processing, has attracted extensive attention from scholars for many years and has been widely used in various practical applications, such as topic labeling (Wang and Manning, 2012), computational phenotyping (Che et al., 2015), question answering (Liu et al., 2015) and dialog act classification (Lee and Dernoncourt, 2016). The performance of text classification relies heavily on the ability of the model to extract textual features from raw text. Previous shallow learning-based text classification approaches mainly use hand-crafted sparse lexical features, such as bag-of-words (BoW) or n-grams, for representing texts (Li et al., 2020). Since these features are predefined, the models do not take full advantage of the large amount of training data. Deep learning architectures represented by convolutional neural networks (CNNs) (Kim, 2014) and recurrent neural networks (RNNs) (Liu et al., 2015) are becoming more popular due to their strong performance in text mining. These models can capture semantic and syntactic information in local consecutive word sequences well.

Recently, graph neural networks (GNNs) have attracted increasing attention due to their superiority in dealing with complex structured data and relations (Kipf and Welling, 2017; Klicpera et al., 2019a). GNNs have achieved promising results in text classification tasks when modeling texts with graph structures due to their powerful expressiveness (Wu et al., 2020). Despite the success of the mentioned models, several serious limitations of prevalent GNNs hinder their performance, which is mainly attributed to the following factors: (I) *Restricted Receptive Fields*. Most previous approaches allow a word in the graph to access direct neighborhoods. In other words, the word embeddings depend solely on the influence of the representation of its neighboring words at each layer. Moreover, the sliding window used to build word-word edges is typically not large. This makes it impossible to enable long-range word interactions. Therefore, it is critical to increase the receptive field of target words to obtain precise text representations. (II) *Shallow Layers*. Most current graph-based models for text classification adopt fairly shallow settings, as they achieve the best performance given two layers. Two-layer graph models aggregate nodes in two-hop neighborhoods and thus cannot extract information beyond two-hop neighbors. Theoretically, we can capture long-range dependencies between words with a large number of layers. However, a common challenge faced by most GNNs is that performance degrades severely when stacking multiple layers to exploit larger receptive fields (Liu et al., 2020). Some researchers attribute this phenomenon to over-smoothing (indistinguishable representation of different classes of nodes) (Li et al., 2018;

*Corresponding author.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 8142–8152
November 7–11, 2021.

Chen et al., 2020). It is still necessary to explore deeper GNNs to obtain more latent features in text classification tasks, especially for short texts, because the available contextual information is limited. (III) *Non-Precision Document-Level Representations*. Most graph-based models for text classification leverage simple pooling operations, such as summing or averaging all nodes in the graph, to obtain document-level representations. This weakens the effect of some key nodes and significantly reduces the expressiveness of the model since different words play distinct roles in the text. (IV) *Low-Pass Filters*. Essentially, existing graph-based methods for text classification are fixed coefficients low-pass filters (Nt and Maehara, 2019). It has been confirmed that the low-pass filter in GNNs mainly preserves the commonality of node features, ignoring the difference among them (Oono and Suzuki, 2020). Therefore, the learned representations of connected nodes become indistinguishable when always adopting a low-pass filter. Meanwhile, several studies have demonstrated the usefulness of high-frequency information in graph signals to enhance the discriminative power of the model especially when the network exhibits disassortativity (Zhu et al., 2020; Chien et al., 2021). Hence, it is necessary to design a filter that does not just exhibit low-pass properties for learning word embeddings.

To overcome the limitations above, we propose a novel model named **D**eep **A**ttention **D**iffusion **G**raph **N**eural **N**etwork (DADGNN) for text classification based on learned effective text representations. Specifically, we use the attention diffusion technique to widen the receptive field of each word in the document, which can capture the long-range word interactions at each layer. Moreover, to extract the deep hidden semantics of words, we decouple the propagation and transformation processes of GNNs to train deeper networks. Finally, we calculate the weight of each node to obtain precise document-level representations. Our work's key contributions are as follows:

(1) We introduce a novel model, namely, DADGNN, based on the attention diffusion and decoupling technique, which has excellent expressive power in modeling documents and overcomes some limitations of conventional graph-based models.

(2) We theoretically prove that the attention diffusion operation is equivalent to a polynomial graph filter that can utilize both high- and low-frequency graph signals.

(3) We conduct extensive experiments on a series of benchmark datasets, and the state-of-the-art performance of DADGNN illustrates its superiority compared to other competitive baseline models.

2 Related Work

2.1 Deep Learning for Text Classification

Compared to traditional text classification methods, deep learning models can automatically learn high-dimensional textual features without the need for tedious feature engineering. Two representative deep learning models, CNNs (Kim, 2014) and RNNs (Liu et al., 2016), have shown powerful capabilities in text classification. To further improve the expressive power of the model, the attention mechanism is introduced as a part of the model, such as in hierarchical attention networks (Yang et al., 2016) and attention-based long short-term memory (LSTM) networks (Wang et al., 2016). However, word-to-word dependencies in the sequential-based learning models above often extend beyond the local sliding window, resulting in lower performance when encoding long sentences.

2.2 Graph Neural Networks

GNNs, as a special type of neural network, have achieved remarkable success in citation networks, social networks and other research areas (Wu et al., 2020). Most of the models above are massage-passing GNNs, including graph convolutional networks (GCNs) (Kipf and Welling, 2017), GraphSAGE (Hamilton et al., 2017) and graph attention networks (GATs) (Velickovic et al., 2018), which aim to learn node embeddings by aggregating information from direct (one-hop) neighbors and stacking multiple layers of information from disjoint (multi-hop) neighbors. More recently, a collection of works introduce a diffusion mechanism in graphs to aggregate information from a larger neighborhood rather than relying only on immediate neighbors, and it capture more complex graph properties, showing powerful performance (Klicpera et al., 2019b; Wang et al., 2020).

Due to the powerful representation capabilities of GNNs, some recent works have used them for text classification tasks (Yao et al., 2019; Wu et al., 2019; Ding et al., 2020). For example, TextGCN (Yao et al., 2019) employs standard GCNs (Kipf and Welling, 2017) on a heterogeneous graph constructed by an entire corpus, obtaining competi-

tive performance. Subsequently, SGC (Wu et al., 2019) removes the additional complexity by iteratively eliminating the nonlinear transformations between GCN layers and collapses the resulting function into a linear transformation, which yields better results. HyperGAT (Ding et al., 2020) proposes to learn text embeddings by applying hypergraphs over documents. However, the aforementioned models utilize only the immediate neighbors in the graph and suffer from over-smoothing issues. Additionally, these models cannot yield high-quality document-level representations by using the sum/mean pooling operation. To the best of our knowledge, our model is the first attempt to utilize the graph attention diffusion method to address the difficulties of long-range word interactions and achieve better performance in text classification.

3 Methods

The overall architecture of DADGNN is shown in Fig.1. Next, we will elaborate on the three aspects of text graph construction, key components and graph-level representation, respectively.

3.1 Text Graph Construction

The initial task of text classification based on GNNs is to represent the serialized text as a graph. In our work, we denote a text as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ indicates a set consisting of different words and $\mathcal{E} = \{e_{1,1}, \ldots, e_{n,n}\}$ indicates the set of edges formed between words. Each node in the graph can initialize a d-dimensional word embedding vector (i.e., word2vec or GloVe). We build an edge starting from a target node and ending at its p-hop adjacent nodes, which is formulized as $e_{i,j}, j \in [i-p, i+p]$, as shown in Fig.2 (a). The advantage of constructing edges in this way is that the graph is directed and its transition matrix is symmetric.

3.2 Key Components

To obtain discriminative feature representations of nodes in deeper networks, we decouple the propagation and transformation processes of GNNs. Concretely, this is formulated as:

$$\begin{aligned} \mathbf{H}^{(0)} &= \mathbf{MLP}(\mathbf{X}) \\ \mathbf{H}^{(l+1)} &= \mathbf{PROPAGATION}(\mathbf{H}^{(l)}) \end{aligned} \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{n \times d}$ represents the original word vector and $\mathbf{H}^{(0)} \in \mathbb{R}^{n \times c}$ represents the vector obtained after feature transformation, in which c is the number of text categories.

The forward propagation process of previous graph-based models can be formulated as:

$$\mathbf{H}^{(l+1)} = \sigma(\tilde{\mathbf{A}}(\ldots(\sigma(\tilde{\mathbf{A}}\mathbf{X}\mathbf{W}^{(0)})\ldots)\mathbf{W}^{(l-1)}) \tag{2}$$

where $\tilde{\mathbf{A}}$ is the adjacency matrix, representing the propagation process of GNNs. $\mathbf{W}^{(\cdot)}$ is a layer-specific learnable weight matrix, and σ is a nonlinear function. These represent the feature transformation process of GNNs.

The entanglement of the propagation and representation transformation in Eq.2 can significantly degrade the performance of GNNs (Liu et al., 2020). In traditional GNNs, the original features need to be propagated and then multiplied by the transformation matrix. Intuitively, it is difficult to train a network when the number of layers becomes large. Because there are so many parameters in transformation intertwined with the receptive field in the propagation. Additionally, propagation and representation transformation affect the graph network in terms of structures and features, respectively. Hence, there is no need to intertwine the two processes. After performing Eq.1, the two fundamental processes of GNNs are decoupled.

Then, we calculate the normalized attention weights between directly connected nodes using Eqs.3 and 4.

$$\mathbf{Y}_{ij}^{(l)} = \begin{cases} a^{(l)}\sigma(\mathbf{W}^{(l)}(\mathbf{H}_i^{(l)} || \mathbf{H}_j^{(l)})) \,, \text{if } e_{i,j} \in \mathcal{E} \\ -\infty \,, \text{otherwise} \end{cases} \tag{3}$$

$$\mathbf{A}^{(l)} = \text{softmax}(\mathbf{Y}^{(l)}) \tag{4}$$

where $\mathbf{W}^{(l)}$ is a weight matrix and $a^{(l)}$ is a weight vector, which are trainable parameters shared by the $l-th$ layer. $\mathbf{A}^{(l)}$ is a graph attention matrix in the $l-th$ layer. Additionally, σ is the ReLU activation function.

In Eq.3 the target node i does not consider the potential impact from the neighbors beyond one hop. We compute the attention between nodes that are not directly connected in a complex network via the diffusion mechanism.

The graph attention diffusion matrix $\mathcal{T}$ is obtained based on the attention matrix $\mathbf{A}$ as follows:

$$\mathcal{T} = \sum_{n=0}^{\infty} \zeta_n \mathbf{A}^n \tag{5}$$

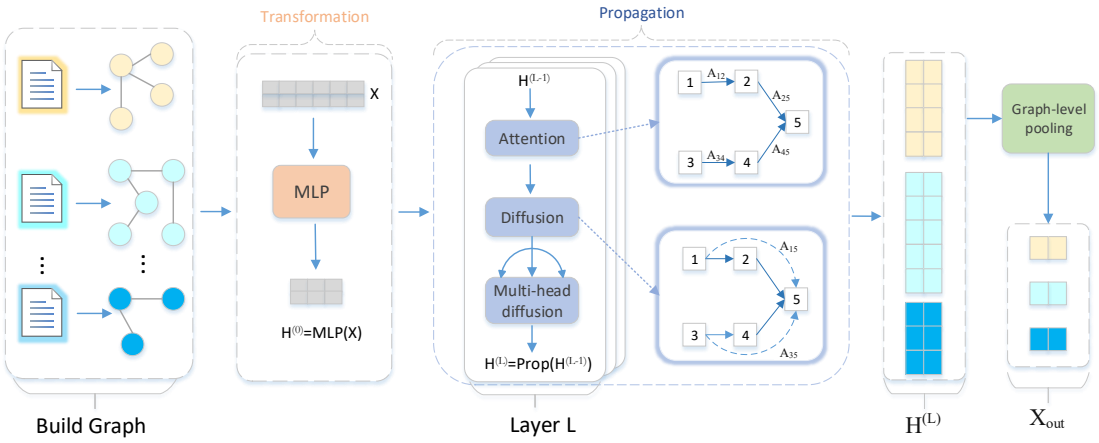
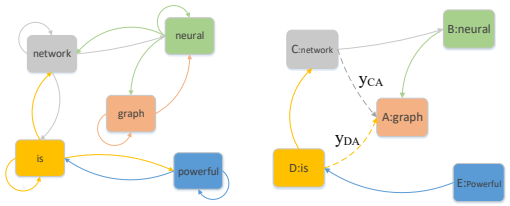


Figure 1: Overall architecture of our model (best viewed in color).



(a) building text graph (b) diffusion procedure

Figure 2: (a) Example of building a text graph, where the text is "graph neural network is powerful" and p is equal to 1. (b) Our model can capture the information of disconnected nodes by considering all paths between nodes via an attention diffusion procedure in a single layer. For example (where our target node is 'graph', and for brevity, we remove the irrelevant edges of (a)), $y_{CA} = \sigma([y_{CB}, y_{BA}])$ (C→B→A) and $y_{DA} = \sigma([y_{DC}, y_{CB}, y_{BA}])$ (D→C→B→A).

where ζ_n are learnable coefficients that depend on the properties exhibited by the constructed graph network. As illustrated in Fig.2 (b), $\mathbf{A}^n$ is the powers of the attention matrix, which take into account the influence of all neighboring nodes j with path lengths up to n on target node i based on the powers of the graph adjacency matrix. This operation effectively increases the attentional receptive field in a single layer of the neural network. This mechanism effectively establishes attentional links between unconnected nodes to obtain the attention coefficients, whose magnitudes depend on ζ_n and the path length.

In practice, according to "four/six degrees of separation" in real-world graphs (Backstrom et al., 2012), which means that the shortest path distance between a pair of nodes is at most four or six, we usually do not sum to $n = \infty$. In our experiments, we truncate the infinite Eq.5 to a natural number $N \in [4, 7]$, i.e., $\mathcal{T} = \sum_{n=0}^{N} \zeta_n \mathbf{A}^n$, which can yield impressive model performance.

To further improve the expressiveness of the attention diffusion layer, we deploy a multi-head attention diffusion mechanism. Specifically, we first compute the attention diffusion for each head k independently and then aggregate them. The output feature representation is as follows:

$$\mathbf{H}^{(l+1)} = [||_{k=1}^{K}(\mathcal{T}\mathbf{H}^{(l)})_k]\mathbf{W}_a \tag{6}$$

where $||$ is the concatenation operation and $\mathbf{W}_a$ denotes a weight matrix for transforming the dimensions.

3.3 Graph-Level Representation

After propagating the $\mathbf{L}$-th layer of our model, we are able to compute the final representations of all nodes on each text graph. To measure the different roles of each node in the graph, in contrast to graph-based text classification models that use general pooling, we employ a node-level attention mechanism. Specifically, it can be expressed via the following equation:

$$\begin{aligned} \Psi &= \text{sigmod}(\mathbf{H}^{(L)}\mathbf{W}_b) \\ \mathbf{X}_{out} &= \sum\nolimits_{i \in \mathcal{V}} \Psi_i \mathbf{H}_i^{(L)} \end{aligned} \tag{7}$$

where $\mathbf{W}_b$ is a trainable weight matrix and Ψ_i denotes the attention coefficient of node i in the graph. To obtain the probability of each category, we further perform $\mathbf{X}_{out} = \mathbf{softmax}(\mathbf{X}_{out})$.

Finally, we use the cross-entropy loss as the objective function to optimize the neural network for text classification. Concretely:

$$Loss = -\sum_{d\in\mathcal{D}}\sum_{\tau}^{c}\Phi_{[d,\tau]}\mathbf{X}_{out[d,\tau]} \tag{8}$$

where $\mathcal{D}$ is the set of training data and Φ is the indicator matrix. Note that our model is directly applicable to inductive learning tasks, and for unseen test documents, the corresponding constructed graph can be fed directly into the trained model for prediction. Moreover, it is trained in an end-to-end manner, which means that the learnable parameters are considered simultaneously when optimizing the network.

4 Spectral Analysis

In this section, we investigate the effectiveness of our model from the perspective of graph spectra.

Graph Fourier Transform. The graph Fourier transform relies on the spectral decomposition of the graph Laplacian. A common symmetric positive semidefinite graph Laplacian matrix is $\mathcal{L} = \mathbf{I} - \mathbf{D}^{-1/2}\hat{\mathbf{A}}\mathbf{D}^{-1/2}$ with eigendecomposition $\mathcal{L} = \mathbf{V}\hat{\Lambda}\mathbf{V}^{\top}$, where $\mathbf{V}$ and $\hat{\Lambda}$ are real-valued, $\mathbf{D}$ is the diagonal degree matrix and $\hat{\mathbf{A}}$ is the adjacency matrix. According to (Shuman et al., 2013), the eigenvectors of $\mathcal{L}$ can be treated as the bases in the graph Fourier transform. The graph Fourier transform acting on the graph signal x is defined as $\hat{x} = \mathbf{V}^{\top}x$, and the inverse graph Fourier transform is $x = \mathbf{V}\hat{x}$. Hence, the graph convolution between the signal x and filter g on $\mathcal{G}$ is defined as

$$\begin{aligned} g *_{\mathcal{G}} x &= \mathbf{V}((\mathbf{V}^{\top}g)\cdot(\mathbf{V}^{\top}x)) = \mathbf{V}\tilde{G}_{\beta}(\hat{\Lambda})\mathbf{V}^{\top}x \\ &= g_{\beta}(\mathcal{L})x \end{aligned} \tag{9}$$

where $\tilde{G}_{\beta}(\hat{\Lambda}) = \mathrm{diag}(g_{\beta,M}(\hat{\lambda}_1),\ldots,g_{\beta,M}(\hat{\lambda}_M))$.

The attention matrix $\mathbf{A}$ is essentially a random walk transition matrix, which satisfies $\mathbf{D}_{ii} = \sum_j^N \mathbf{A}_{ij} = 1$ and $\mathbf{A}_{ij} > 0$. Therefore, the symmetric normalized graph Laplacian matrix can be denoted as $\mathcal{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{A}$. A filter g_{β} acting in a Laplacian matrix can be appreciated as a polynomial filter of order M, i.e., $g_{\beta}(\mathcal{L}) = \sum_{m=0}^{M}\beta_m\mathcal{L}^m$, since it is localized and shift invariant (Defferrard et al., 2016). Since $\mathcal{T} = \sum_{n=0}^{N}\zeta_n\mathbf{A}^n$, inspired by (Klicpera et al., 2019b), we can obtain the relationship between the graph filter and the graph attention diffusion using the binomial theorem as

$$\begin{aligned} \sum_{m=0}^{M}\beta_m\mathcal{L}^m &= \sum_{m=0}^{M}\beta_m(\mathbf{I}-\mathbf{A})^m \\ &= \sum_{m=0}^{M}\beta_m\sum_{n=0}^{m}\binom{m}{n}\mathbf{I}^{m-n}(-\mathbf{A})^n \\ &= \sum_{m=0}^{M}\beta_m\sum_{n=0}^{m}\binom{m}{n}(-1)^n\mathbf{A}^n \\ &= \sum_{n=0}^{M}\sum_{m=n}^{M}\beta_m\binom{m}{n}(-1)^n\mathbf{A}^n \end{aligned} \tag{10}$$

The last equation holds because $m, n \in [0, M]$ and $n \le m$. If we set $\zeta_n = \sum_{m=n}^{M}\beta_m\binom{m}{n}(-1)^n$ and $M = N$, we can view the graph attention diffusion $\mathcal{T}$ as a polynomial graph filter. Notably, polynomial graph filters can approximate any graph filter (Shuman et al., 2013), and a good choice of ζ_n can capture both the node information and the graph structure. Since ζ_n is a learnable parameter, the ideal ζ_n value represents the optimal graph filter.

Given the graph attention matrix $\mathbf{A} = \mathbf{U}\Lambda\mathbf{U}^{\top}$, via the eigenvalue decomposition of $\mathbf{A}$, we can define the attention diffusion matrix $\mathcal{T}$ acting on the graph signal x as

$$\begin{aligned} \mathcal{T}x &= g_{\zeta,N}(\mathbf{A})x = g_{\zeta,N}(\mathbf{U}\Lambda\mathbf{U}^{\top})x \\ &= \mathbf{U}g_{\zeta,N}(\Lambda)\mathbf{U}^{\top}x = \mathbf{U}(\sum_{n=0}^{N}\zeta_n\Lambda^n)\mathbf{U}^{\top}x \end{aligned} \tag{11}$$

where $g_{\zeta,N}(\Lambda) = \mathrm{diag}(g_{\zeta,N}(\lambda_1),\ldots,g_{\zeta,N}(\lambda_n))$ and the corresponding filter coefficients are $g_{\zeta,N}(\lambda) = \sum_{n=0}^{N}\zeta_n\lambda^n$, whose λ corresponds to the eigenvalue of $\mathbf{A}$. By limiting the parameter ζ_n, we can obtain graph filters with different characteristics, such as low-pass, high-pass and even more complex filters.

In recent works, GDC (Klicpera et al., 2019b) and MAGNA (Wang et al., 2020), $g_{\zeta,N}(\cdot)$ acts a low-pass graph filter by setting stricter conditions, i.e., $\zeta_n = \alpha(1-\alpha)^n$, $\sum_{n=0}^{N}\zeta_n = 1$ and $\alpha \in (0,1)$. This enhances low-frequency information corresponding to the large-scale graph structure and suppresses high-frequency information corresponding to the fine-grained message. However, as mentioned in Section 1, the high-frequency information of a graph is not always useless, and it can

capture the difference between nodes. Eliminating them completely may limit the expressiveness of the model. When $\zeta_n = (-\alpha)^n$ and $\alpha \in (0, 1)$, $g_{\zeta,n}(\cdot)$ is a high-pass filter (Chien et al., 2021). Since our model can adaptively learn the optimal weights ζ_n rather than behave only as a low-pass filter, DADGNN is theoretically more expressive than fixed-weight filters, and the results in Table 2 strongly support our inference.

5 Experiments

	Long Corpus					Short Corpus				
Dataset	IMDB	WebKB	R52	R8	AG_news	DBLP	TREC	MR	SST-1	SST-2
#Doc	50,000	4,199	9,100	7,674	127,600	81,479	5952	10,662	11,855	9,613
#Train	25,000	2,803	6,532	5,485	120,000	61,479	5452	7,108	9,465	7,792
#Test	25,000	1,396	2,568	2,189	7,600	20,000	500	3,554	2,210	1,821
#Word	71,278	7,771	8,892	7,688	128,515	25,549	9593	18,764	19,524	17,561
Avg.Length	232.77	133.37	69.82	65.72	44.03	8.51	10.06	20.39	20.17	20.32
#Class	2	7	52	8	4	6	6	2	5	2

Table 1: Summary statistics of evaluation datasets.

Datasets. For a fair and comprehensive evaluation, we select two types of text corpora, which include both long and short documents from different domains. The detailed summary statistics of the datasets are presented in Table 1.

Long Corpus. (1) AG news (Zhang et al., 2015) is a subdataset of AG's corpus of news articles constructed by assembling titles and description fields from the 4 largest classes. (2) IMDB (Maas et al., 2011) is a movie review dataset for sentiment classification. We follow the data processing method used in (Tang et al., 2015). (3) WebKB (Craven et al., 1998) is a dataset that includes web pages from the computer science departments of various universities categorized into 6 imbalanced categories. (3) Reuters is a collection dataset of news documents. We use two subdatasets, R8 and R52, in our experiment, following the settings of (Yao et al., 2019).

Short Corpus. (1) MR (Pang and Lee, 2005), a movie review dataset, is specifically for binary sentiment analysis, and each review only contains one sentence. (2) SST-1 (Socher et al., 2013) is a fine-grained sentiment analysis dataset in which each sentence is annotated with fine-grained labels (from very positive to very negative). (3) SST-2 (Socher et al., 2013) is the same as SST-1 except that neutral reviews are removed, and the data are labeled as positive or negative. (4) TREC (Li and Roth, 2002) is a question dataset; the task involves classifying questions into 6 question types. (5) DBLP is a computer science bibliography dataset that contains the titles of papers. Six different research areas are selected. We use the same data as (Tang et al., 2015).

Baselines. We divide the baseline models into three categories for comparison: (I) sequence-based deep learning models, such as CNNs (Kim, 2014), LSTMs (Liu et al., 2016) and bidirectional LSTM (Bi-LSTM) (Huang et al., 2015); (II) word embedding-based models, mainly including fastText (Joulin et al., 2017), PV-DM (Le and Mikolov, 2014) and LEAM (Wang et al., 2018); and (III) graph-based models, such as Graph-CNN (Defferrard et al., 2016), TextGCN (Yao et al., 2019), SGC (Wu et al., 2019), Text-level GNN (Huang et al., 2019) and HyperGAT (Ding et al., 2020). Notably, HyperGAT is a state-of-the-art baseline that captures text representation on document-level hypergraphs.

Implementation Details. In our experiment, we use the same data preprocessing as (Yao et al., 2019). Concretely, we remove the words that occur fewer than 5 times and clean and tokenize the raw documents. Moreover, we randomly sample 10% of the training data as the validation set. The optimal values of the hyperparameters are selected when the model reaches the highest accuracy of the validation samples. A pre-training GloVe word embedding (Pennington et al., 2014) is adopted for those models that require word initialization. We train DADGNN for 200 epochs with an early-stopping strategy using the Adam method (Kingma and Ba, 2015). For the baseline models, we use the default hyperparameters described in the original paper. All experiments are conducted ten times to obtain the average accuracy and standard deviation.

6 Results and Analysis

Table 2 shows that DADGNN has the state-of-the-art performance in all datasets compared to other competitive baselines, which shows the superiority of our model. Additionally, we obtain the following insights and analysis below.

One crucial reason why our model achieves more significant improvements is that the receptive field of the target node is enhanced by attention diffusion, which incorporates more informative messages (i.e., both low-frequency and high-frequency information) in the text. Furthermore, it is possible to obtain distinguishable hidden features in deep graph networks by decoupling the procedures of feature transformation and propagation, which is

Model	IMDB	WebKB	R52	R8	AG news	DBLP	TREC	MR	SST-1	SST-2
CNN	86.15±0.60	86.87±0.23	87.59±0.48	95.71±0.52	89.13±0.31	75.28±0.61	93.62±0.55	77.75±0.72	42.30±0.41	80.07±0.51
LSTM	85.91±0.71	86.51±0.77	90.48±0.86	96.09±0.19	86.06±0.72	74.11±0.75	93.01±0.41	77.33±0.89	41.92±0.63	79.52±0.61
Bi-LSTM	86.62±0.16	86.57±0.36	90.54±0.91	96.31±0.33	86.52±0.31	72.25±1.27	93.32±0.72	77.68±0.86	42.63±0.66	80.56±0.21
fastText	80.21±0.25	82.96±0.36	92.81±0.09	96.13±0.21	91.49±0.12	71.19±0.52	91.29±0.69	75.14±0.20	36.08±0.81	81.45±0.16
PV-DBOW	75.96±0.26	72.62±0.41	78.29±0.11	85.87±0.10	81.25±0.36	63.59±0.21	80.36±0.35	61.09±0.10	38.12±0.33	72.92±0.12
LEAM	83.29±0.55	83.95±0.25	91.84±0.23	93.31±0.24	91.75±0.35	72.62±1.59	89.21±0.57	76.95±0.45	42.93±0.69	80.52±0.19
Graph-CNN	OOM	83.29±1.22	92.75±0.22	96.99±0.12	87.56±0.29	71.37±1.26	90.39±1.52	77.22±0.27	35.23±0.21	76.95±0.62
TextGCN	OOM	86.17±0.96	93.56±0.18	97.07±0.10	90.84±1.32	76.72±0.69	91.40±0.39	76.74±0.20	40.65±0.06	81.02±0.40
SGC	OOM	87.39±0.66	94.02±0.21	97.21±0.11	91.06±0.62	76.79±0.72	92.29±1.26	75.91±0.36	41.63±0.41	75.95±0.92
Text-level GCN	OOM	89.91±0.51	94.62±0.32	97.83±0.20	OOM	OOM	94.09±0.36	75.96±0.56	43.02±0.65	81.75±0.36
HyperGAT	86.32±0.71	87.46±0.55	94.98±0.27	97.97±0.23	91.24±0.56	72.56±0.96	93.55±1.79	78.32±0.27	41.96±0.35	81.26±0.72
DADGNN(ours)	**88.49±0.59**	**90.92±0.42**	**95.16±0.22**	**98.15±0.16**	**92.24±0.36**	**78.59±0.62**	**97.99±0.52**	**78.64±0.29**	**45.15±0.26**	**84.32±0.15**

Table 2: The results of test accuracy on document classification with different models. For each model, the mean ± standard deviation is reported. DADGNN significantly outperforms all the baselines based on t-tests ($p < 0.05$). Underline: runner-up. OOM:>16 GB.

extremely useful for short texts.

Most graph-based approaches achieve outstanding performance compared to the other two groups of baseline models in the topic classification datasets, such as R52, R8 and AG news. This phenomenon demonstrates the critical role of capturing non-consecutive and long-distance semantics in the document for text classification. However, for the sentiment classification task, the sequence-based models have more robust performance, which can be explained by the fact that word order modeling is crucial for capturing sentiment. Since our model constructs directed graphs, it can obtain sequential context information ignored by most graph-based methods and achieves excellent results in sentiment classification.

Ablation Study. To verify the role of each module in DADGNN, we perform a series of ablation experiments, and the results are shown in Table 3. From the results, we can see that removing any part of our proposed model leads to a decrease in accuracy. In particular, removing the attention diffusion module degrades performance most significantly, indicating the effectiveness of attention diffusion for increasing the receptive field to learn more expressive word representations. Moreover, using the node-level attention mechanism can further enhance the performance of our approach because of the ability to obtain more precise graph-level representations. It is clear that stacking multiple layers captures long-distance and non-consecutive semantics and thus performs better than the one-layer model.

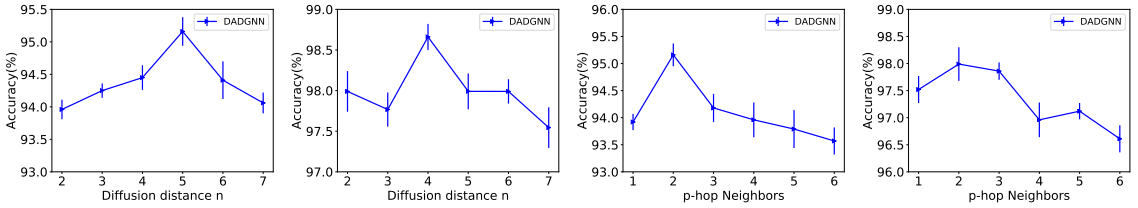
Memory Consumption. We select two representative models for GPU memory consumption analysis to verify the computational efficiency of our model. TextGCN is a transductive model, and the others are inductive models. From the results in Table 4, we can conclude that inductive models consume significantly less memory than transductive models. Our approach consumes less memory with better performance than the previous most computationally efficient HyperGAT model. An important reason for this is that we transform representations into a low-dimensional space by feature transformation at an early stage, making DADGNN computationally efficient and memory-saving.

Hyperparameter Sensitivity. We investigate the impact of different hyperparameters on model performance. Fig.3 (a) and (b) illustrate the effect of the diffusion distance n on model performance in Eq.5 on R52 and TREC datasets, respectively. We observe that the two datasets share an almost consistent variation curve; i.e., the test accuracy first increases and then decreases, with the optimal $n \in [4, 6]$, which is consistent with the previous analysis in Section 3.2. Fig.3 (c) and (d) show the results of test accuracy with different p-hop adjacent neighbors constructing the text graph in Section 3.1 on R52 and TREC datasets, respectively. We find that our proposed model achieves optimal performance when connecting two neighbors. This implies that the nodes cannot understand dependencies across multiple words in the context of connecting only with direct neighbors. However, when connecting to more distant neighbors, the graphs become similar but ignore local features.

Deeper Layers. To verify whether DADGNN stacking multiple layers suffers from the over-smoothing problem common to GNNs, we conduct some experiments under different model depths than those of classical GNNs models, such as TextGCN and SGC. The results are presented in Fig.4. Overall, the accuracy of our model is stable and does not show a significant decrease with the increasing number of layers, which suggests

Model	IMDB	WebKB	R52	R8	AG news	DBLP	TREC	MR	SST-1	SST-2
w/o attention diffusion	86.85±0.61	90.10±0.25	92.66±0.29	96.15±0.35	91.71±0.19	77.13±0.51	95.32±0.22	77.87±0.27	44.67±0.17	83.20±0.25
w/o node-level attention	87.63±0.56	90.45±0.52	93.78±0.17	97.38±0.16	91.78±0.37	78.41±0.39	96.65±0.26	78.43±0.24	43.47±0.12	83.76±0.32
DADGNN (1 layer)	87.99±0.39	90.76±0.39	93.98±0.49	97.79±0.22	91.82±0.21	78.31±0.46	97.10±0.42	78.51±0.13	44.99±0.27	84.04±0.09
DADGNN	88.49±0.59	90.92±0.42	95.16±0.22	98.15±0.16	92.24±0.36	78.59±0.62	97.99±0.52	78.64±0.29	45.15±0.26	84.32±0.15

Table 3: The results of ablation studies on our model based on t-tests($p < 0.05$). Concretely, *w/o attention diffusion* is a variant that adopts the attention of direct neighbors without attention diffusion. Additionally, *w/o node-level attention* is a variant that replaces node-level attention graph pooling with simple sum pooling.

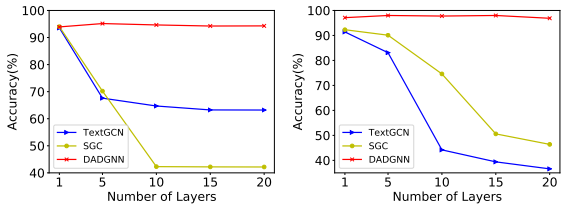


(a) R52 (b) TREC (c) R52 (d) TREC

Figure 3: Hyperparameter sensitivity of our model. Other datasets show the same trend, omitted for space.

Model	TextGCN	HyperGAT	DADGNN
IMDB	OOM	297.96MB	**186.21MB**
WebKB	2172.79MB	92.56MB	**35.97MB**
R52	1289.48MB	46.85MB	**25.21MB**
R8	931.58MB	41.75MB	**13.78MB**
AG news	7362.56MB	206.76MB	**131.19MB**
DBLP	2498.56MB	55.49MB	**29.35MB**
TREC	261.52MB	23.86MB	**10.32MB**
MR	3338.24MB	80.99MB	**32.32MB**
SST-1	665.16MB	68.51MB	**30.52MB**
SST-2	389.12MB	65.17MB	**28.85MB**

Table 4: Comparison of GPU memory consumption of different models. The same hardware condition is remain for conducting all experiments. OOM:>16 GB.



(a) R52 (b) TREC

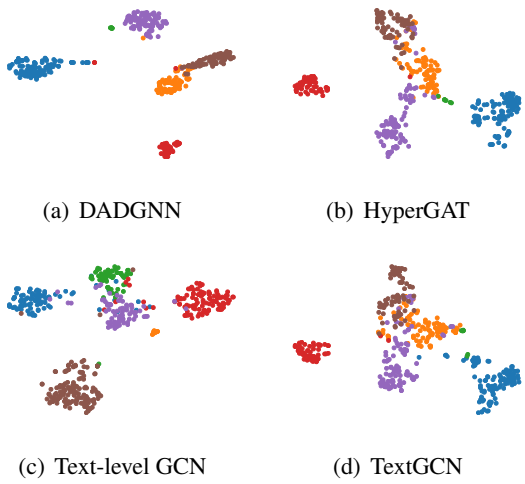
Figure 4: Accuracy with different numbers of layers. Other datasets show the same trend, omitted for space.

that DADGNN has a good ability to mitigate the over-smoothing issue. However, it can be clearly seen that the performace of classical GNNs drops rapidly when stacking multiple layers, indicating that a serious over-smoothing issue is encountered. An important reason for this is that, as mentioned in Section 3.2, transformation representation and propagation are decoupled in GNNs, which benefit from a deep model architecture. The other reason is that we design the graph filter to not only exhibit a low-pass property but to prevent node features from becoming indistinguishable when more high-frequency information is introduced.

Visualization. To show the superiority of representations learned by our model, we use t-SNE (Van der Maaten and Hinton, 2008) method to visualize the learned embeddings of test documents. Concretely, we select three powerful baselines (i.e. HyperGAT, Text-level GCN and TextGCN) to compare with DADGNN in TREC dataset. From Fig.5, we observe that our model is able to learn more discriminative and distinguishable document embeddings over other methods.

7 Conclusion

In this work, we propose a new graph-based model, named DADGNN, for text classification. Our model decouples the necessary procedures of GNNs (i.e., representation transformation and propagation) to train a deep neural network and introduces a attention diffusion mechanism to capture non-direct-neighbor context information in a single layer. Furthermore, the node-level attention technique is introduced to obtain more precise document-level representations. Using the techniques above, our model enhances the receptive field and increases the depth without suffering from over-smoothing problems. The theoretical analysis essentially shows that DADGNN can learn the optimal filter to adapt the dataset; i.e., it can preserve more relevant high-frequency information of



(a) DADGNN (b) HyperGAT

(c) Text-level GCN (d) TextGCN

Figure 5: The t-SNE visualization of different models for test documents on TREC.

nodes to further improve the expressiveness of the model. Extensive experiments show that our model is much better than other competitive approaches. More importantly, our work not only provides a powerful baseline model for text classification but also contributes to graph representation learning.

Acknowledgments

The authors are grateful for the support of the National Natural Science Foundation of China [Grants 61972174 and 62172187], the Science-Technology Development Plan Project of Jilin Province [Grants 20190303006SF, 20190302107GX, 20200801033GH, and YDZJ202101ZYTS128], the Science and Technology Planning Project of Guangdong Province [Grant 2020A0505100018], Guangdong Key-Project for Applied Fundamental Research [Grant 2018KZDXM076]. Fausto Giunchiglia's work is funded by National Project PRIN 2017 "Delphi".

References

Lars Backstrom, Paolo Boldi, Marco Rosa, Johan Ugander, and Sebastiano Vigna. 2012. Four degrees of separation. In *Proceedings of the Annual ACM Web Science Conference*, pages 33–42.

Zhengping Che, David Kale, Wenzhe Li, Mohammad Taha Bahadori, and Yan Liu. 2015. Deep computational phenotyping. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 507–516.

Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. 2020. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3438–3445.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. 2021. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations*.

Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom M. Mitchell, Kamal Nigam, and Seán Slattery. 1998. Learning to extract symbolic knowledge from the world wide web. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference*, pages 509–516.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*.

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*, pages 4927–4936.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034.

Lianzhe Huang, Dehong Ma, Sujian Li, Xiaodong Zhang, and WANG Houfeng. 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*, pages 3435–3441.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Armand Joulin, Édouard Grave, Piotr Bojanowski, and Tomáš 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*, pages 427–431.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019a. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations*.

Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. 2019b. Diffusion improves graph learning. In *Advances in Neural Information Processing Systems*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196.

Ji Young Lee and Franck Dernoncourt. 2016. Sequential short-text classification with recurrent and convolutional neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 515–520.

Qian Li, Hao Peng, Jianxin Li, Congyin Xia, Renyu Yang, Lichao Sun, Philip S Yu, and Lifang He. 2020. A text classification survey: From shallow to deep learning. *arXiv preprint arXiv:2008.00364*.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *The 19th International Conference on Computational Linguistics*.

Meng Liu, Hongyang Gao, and Shuiwang Ji. 2020. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 338–348.

Pengfei Liu, Xipeng Qiu, Xinchi Chen, Shiyu Wu, and Xuan-Jing Huang. 2015. Multi-timescale long short-term memory neural network for modelling sentences and documents. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2326–2335.

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*, pages 2873–2879.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 142–150.

Hoang Nt and Takanori Maehara. 2019. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*.

Kenta Oono and Taiji Suzuki. 2020. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 115–124.

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*, pages 1532–1543.

David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1165–1174.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11).

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Guangtao Wang, Rex Ying, Jing Huang, and Jure Leskovec. 2020. Direct multi-hop attention based graph neural network. *arXiv preprint arXiv:2009.14332*.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2321–2331.

Sida I Wang and Christopher D Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 90–94.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International Conference on Machine Learning*, pages 6861–6871.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7370–7377.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond homophily in graph neural networks: Current limitations and effective designs. In *Advances in Neural Information Processing Systems*.