GAINER: Graph Machine Learning with Node-specific Radius for Classification of Short Texts and Documents

Naganand Yadati
National University of Singapore
naganand@nus.edu.sg and y.naganand@gmail.com

Abstract

Graphs provide a natural, intuitive, and holistic means to capture relationships between different text elements in Natural Language Processing (NLP) such as words, sentences, and documents. Recent advancements in the field of Graph Machine Learning (GML) have led to the development of numerous models to process text for various natural language applications, including but not limited to short-text classification, document classification, and others. At the heart of GML models, specifically those based on Graph Neural Networks (GNNs), lies the message passing operation which has shown to be an essential component for strong empirical performance in NLP. However, the number of message passing steps (often known as the radius) is *fixed for all the nodes* in existing GML models for NLP. Fixing the radius poses a fundamental restriction as nodes exhibit diverse properties and varying amounts of informative local structures in the input graph. This paper presents GAINER, a novel framework called Graph mAchine learnIng with Node-spEcific Radius, aimed at graph-based NLP. We propose non-neural and novel neural approaches built on the core ideas of GAINER. Through rigorous experimentation, we demonstrate the efficacy of GAINER in popular NLP tasks.

1 Introduction

Graphs present a natural, intuitive, and holistic representation for understanding the interactions that exist among different text elements, such as words, sentences, and documents. The use of graphs provides a wide array of options for effectively representing and tackling different problems in Natural Language Processing (NLP). For instance, world-level, sentence-level, and document-level graphs capture various aspects of text datasets. Recent breakthroughs in Graph Machine Learning (GML), notably driven by the progress made in Graph Neural Networks (GNNs) (Wu et al., 2022, 2021; Ma and Tang, 2020), have led to the development of numerous models tailored for processing text. Diverse NLP applications span a wide range (Liu and Wu, 2022), including but not limited to short-text classification and document classification

At the core of GNNs, the message passing operation (Gilmer et al., 2017) plays a pivotal role in achieving remarkable success in NLP (Wu et al., 2023). However, in popular GNN models, the number of message passing steps, often known as the radius, is predetermined and remains fixed for every node in the input graph. For instance, in a three-hop GNN, each node gathers information from nodes that are within a three-hop radius. Fixing the number of hops (i.e. radius) poses a fundamental restriction as nodes exhibit diverse properties and varying amounts of informative local structures in the input graph. In an intuitive sense, nodes with poor connectivity tend to derive greater advantages from a higher radius, whereas well-connected nodes may require only a limited radius. A GNN with a very small radius may not propagate enough information, resulting in limited smoothing effects for certain nodes. On the other hand, a GNN with a very large radius may oversmooth the information (Rusch et al., 2023), leading to reduced node-specific characteristics.

The prevalent approach in GNNs for NLP research, including very recent publications (Liu et al., 2023; Zheng et al., 2022), involves the application of a 2-layer Graph Convolutional Network (Kipf and Welling, 2017). While this method performs adequately for nodes with strong connections, it struggles with nodes having limited or weak connections, such as low-degree nodes connected to other low-degree nodes.

Inspired by the aforementioned fundamental limitations of existing GML models in graph NLP, our work makes the following contributions:

- We propose GAINER, a novel framework called Graph mAchine learnIng with Node-

Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 609–626
March 17-22, 2024 ©2024 Association for Computational Linguistics

spEcific Radius, aimed at graph NLP (Please see Figure 1 and Section 4).

- We propose novel approaches aimed at graph NLP, comprising Simple-GAINER (a non-neural approach) and Neural-GAINER, built upon the core idea of GAINER (Please see Sections 4.3 and 4.6).

- We demonstrate the adaptability of GAINER and its efficacy in a wide range of tasks including short-text classication, document classification on text attributed graphs, and document coherence assessment. Our methods achieve statistically significant results on 5 of the 6 datasets evaluated (Please see Section 5).

2 Related Work

We divide the related work into three subsections.

2.1 Graph Machine Learning (GML)

The prevailing trend in machine learning models for graph-structured inputs involves the learning of representations for graph nodes (Hamilton, 2020). Many of these models are built upon GNNs (Wu et al., 2022; Ma and Tang, 2020) and message passing neural networks (Gilmer et al., 2017). GNNs such as graph convolutional networks (Kipf and Welling, 2017), Graph Sample and AGgregatE (Hamilton et al., 2017), graph attention networks (Veličković et al., 2018), and graph isomorphism networks (Xu et al., 2019) have gained immense popularity in the field. Simplified graph convolution (Wu et al., 2019) offers an effective linearised model for GML that eliminates non-linear activations found in vanilla GCNs. This development has inspired the emergence of linear graph convolutions in the current literature (Zhu and Koniusz, 2021; Huang et al., 2021; Abu-El-Haija et al., 2021; Wang et al., 2021b; Zhang et al., 2021, 2022b).

2.2 Relevant Breakthroughs in GNNs

Decoupled GNNs, characterised by the separation of the message passing operation and the feature transformation operation, have emerged as effective models in GML tasks (Dong et al., 2021; Chien et al., 2021; Chen et al., 2020; Bojchevski et al., 2020; Klicpera et al., 2019). These models have recently showcased competitive performances, highlighting the effectiveness of decoupling the two key operations.

Adaptive GNNs, equipped with gate/attention mechanisms or reinforcement learning, have been suggested by numerous learning-based approaches to dynamically aggregate information for each individual node (Huang et al., 2023; Ma et al., 2021; Spinelli et al., 2021; Miao et al., 2021; Lai et al., 2020). However, these methods bring about increased training complexity and a lack of interpretability, thus constraining their applicability.

Our proposed method merges the strengths of decoupled and adaptive approaches, offering a blend of simplicity and adaptability tailored to task-specific applications.

2.3 GNNs in NLP

The presence of graph structures in a wide range of NLP problems has sparked a surge of interest in utilising GNNs as a promising approach to tackle several NLP tasks effectively (Liu and Wu, 2022). GNNs were initially employed on syntactic dependency trees to learn syntax-aware latent feature representations for words in sentences. Graph Convolutional Networks (GCNs) were used specifically to enhance the performance of tasks like Semantic Role Labeling (Marcheggiani and Titov, 2017) and Machine Translation (Bastings et al., 2017). In subsequent developments, GNNs have been successfully employed in a range of NLP tasks beyond their initial applications, including relation extraction (Xu and Choi, 2022; Nguyen et al., 2022), question answering (Wang et al., 2023; Zhang et al., 2022a), knowledge graphs (Li et al., 2023b), summarisation (Qiu and Cohen, 2022; Chen et al., 2022), and many more. Among the numerous publications, there exists a subset of works that specifically address tasks involving graphs in the context of text classification and document processing (Liu et al., 2023; Li et al., 2023a; Zheng et al., 2022).

In most of the existing literature on GNNs in NLP, a 2-layer GCN is commonly employed, which may work well for nodes with strong connections but falls short in effectively handling nodes with weak connections in the graph (e.g., low degree nodes connected to other low degree nodes). Our proposed idea of employing a node-specific radius is specifically tailored to tackle nodes characterised by a weak or inadequate connectivity structure. In this study, we investigate text classification and document processing tasks as illustrative examples and leave other tasks for future work.

3 Preliminaries

We present notation to introduce the method and discuss problems studied in the paper.

3.1 Notations Used

We first delve into the notations used in this work, to establish a common understanding of the symbols and terminology used throughout the paper.

Input Graph: Let $G = (V, E)$ be an input undirected graph where $V = \{1, 2, \cdots, n\}$ is a set of n nodes and $E \subseteq V \times V$ is a set of edges. Let $\tilde{\mathbf{A}} \in \{0, 1\}^{n \times n}$ be the adjacency matrix of G with self-loops, i.e., $\tilde{A}_{v,v} = 1$ for all $v \in V$. Note that $\tilde{A}_{v,u} = 1$ if and only if there exists an edge betwneen $v \in V$ and $u \in V$. Let $\mathbf{\Delta}$ be a diagonal matrix consisting of the node degrees, i.e., $\Delta_{v,v} = \sum_{u=1}^{n} \tilde{A}_{v,u}$ and zero entries elsewhere. We assign the symbol A to represent the symmetrically normalised adjacency matrix $\mathbf{A} = \mathbf{\Delta}^{-\frac{1}{2}} \tilde{\mathbf{A}} \mathbf{\Delta}^{-\frac{1}{2}}$.

Node Features: Each node $v \in V$ is associated with a $d-$dimensional input feature vector $\mathbf{x_v} \in \mathbb{R}^d$. The matrix $\mathbf{X^{(0)}} = [\mathbf{x_1} \cdots \mathbf{x_n}]^T \in \mathbb{R}^{n \times d}$ denotes the input feature matrix. The superscript 0 in $\mathbf{X^{(0)}}$ signifies that the features utilised in the GML model are not treated as hidden features but are instead directly incorporated as input.

3.2 Graph Convolutional Network (GCN)

Many problem instances in Graph NLP are approached through the popular GCN model (Kipf and Welling, 2017) as a go-to solution, capitalising on its ability to integrate the graph G and the input node features $\mathbf{X^{(0)}}$. Leveraging an aggregation process, the GCN model merges the features of a node with the features of its neighbours, enabling the creation of smoother node representations. The process of an L- layer GCN can be defined as

$$\mathbf{X}^{(l+1)} = \eta \left(\mathbf{A} \mathbf{X}^{(l)} \mathbf{W}^{(l)} \right), \; l = 0, \cdots, L-1, \quad (1)$$

where $\eta(\cdot)$ is the activation function and $\mathbf{W}^{(l)}$ is a layer-specific trainable weight matrix at layer l.

3.3 Example Contexts

Within the scope of this paper, we analyse noteworthy NLP problems, drawing attention to the nodes and edges of the input graph $G = (V, E)$, and the node features $\mathbf{X^{(0)}}$ exploited by GNNs in NLP.

1) Short Text Classification: Based on recent research (Zheng et al., 2022; Wang et al., 2021a), graphs have played a crucial role in improving classification of short texts. Nodes of the input graph could represent words in short texts, in which case the input node features could be pre-trained word embeddings, e.g., GloVe (Pennington et al., 2014). Edges in such a graph could capture relationships between words that have notable co-occurrences in a large corpus, quantified by metrics such as point-wise mutual information.

2) Document Classification in Text Attributed Graphs: In the domain of text attributed graphs, the customary practice involves using nodes to represent documents for node classification purposes (He et al., 2023; Zhang et al., 2018). The input node features capture specific characteristics of the documents, such as their title and abstract, encoded by embeddings (either pre-trained, trainable, or hand-crafted). Citation links between documents, acting as undirected edges, naturally connect two similar documents and are utilised by GNNs.

3) Document Coherence Assessment: An alternative way to model the structural similarity of documents is by analysing the sentences within them (Guinaudeau and Strube, 2013), which has particularly been valuable for coherence assessment. Sentences are represented by nodes, and node features are obtained through pre-trained embeddings of language models. The existence of an edge between two structurally similar sentences is determined by the strong semantic relations among the nouns in those sentences (Liu et al., 2023).

4 Proposed Framework: GAINER

In the aformentioned examples, the existing literature employs an L-layer GCN, which considers L-hop information around each node to propagate and smooth information across edges. The number of layers L, is considered a hyperparameter, and empirical results suggest that setting $L = 2$ yields the best performance in most cases. Figure 1 visually illustrates the primary contribution of GAINER and highlights the distinctions from 2-layer GCNs, that are commonly used.

4.1 Motivation

While the approach proves effective for well-connected nodes, such as (i) high-degree nodes, or (ii) low-degree nodes with high-degree neighbours, it falls short when it comes to poorly connected nodes, such as low-degree nodes connected to other low-degree nodes. Furthermore, as the value of

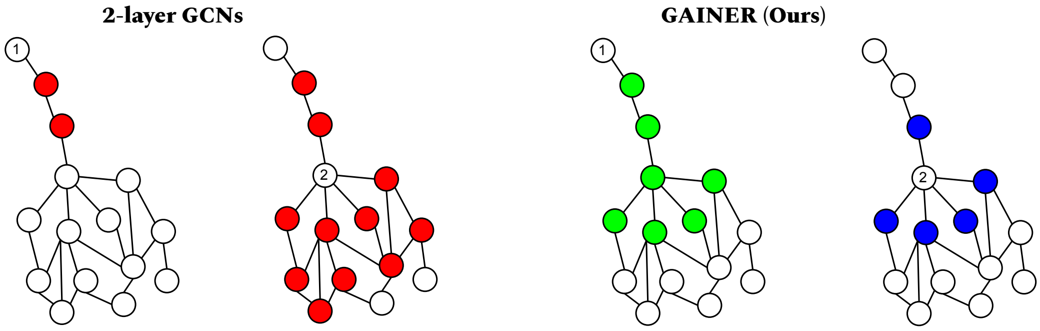


Figure 1: (Best seen in colour) Illustrating the difference between existing 2-layer GCNs and the proposed GAINER. The graph is the same in all the four images. In the first and third images, the node of interest is indexed by 1, while in the second and fourth images, it is the node with index 2. A GCN with only 2 layers might not capture information from a sufficient number of hops, leading to an inadequate representation of poorly-connected nodes (first image). By examining the second image, we can see that a well-connected node possesses a 2-hop neighborhood that spans a significant portion of the graph. Adding more GCN layers can lead to excessive smoothing, resulting in highly similar representations for the majority of nodes. Third and fourth images illustrate that by incorporating a node-specific radius, GAINER can flexibly adjust the degree of smoothing, leading to larger radii for poorly connected nodes (e.g., green nodes around 1) and smaller radii for well-connected nodes (e.g., blue nodes around 2).

L increases, the hidden representations of well-connected nodes become excessively smoothed, resulting in oversmoothing (Rusch et al., 2023).

Our proposed approach to address this tradeoff revolves around the introduction of a node-specific radius, represented as $r(v, \tau)$, as a replacement for the conventional number of layers L in GCNs. This radius is assigned to each node $v \in V$, and is complemented by a threshold value $\tau > 0$. This motivates our framework referred to as GAINER [1] (Graph machIne learnIng with Node-spEcific Radius), which forms the basis of our approaches.

4.2 Simplifying the GCN Process

An essential finding in the GCN process, as described by Equation 1, is that when the activation function $\eta(\cdot)$ is the identity function and $\mathbf{W}^{(l)}$ are identity matrices for $l = 1, \cdots, L-1$, the resulting model is the simplified graph convolution (SGC) model (Wu et al., 2019) given by

$$\mathbf{X}^{(L)} = \mathbf{A}^{L}\mathbf{X}^{(0)}\mathbf{W}^{(0)}. \tag{2}$$

SGC has emprically shown to be highly competitive in terms of accuracy and offers substantial training speed improvements over GCN across various datasets, including NLP datasets. It is important to note that in Equation 2, the notation $\mathbf{A}^{L}$ represents the matrix $\mathbf{A}$ raised to the power of L.

4.3 Simple-GAINER (SGR)

The essence of GAINER becomes evident when we examine Equation 2 on a per-node basis, replacing L with $r(v, \tau)$ for each node $v \in V$ in the graph:

$$\mathbf{X}_v^{(r(v,\tau))} = [\mathbf{A}^{r(v,\tau)}\mathbf{X}^{(0)}\mathbf{W}^{(0)}]_v. \tag{3}$$

Equation 3 employs the notation $[\mathbf{M}]_v$ to represent the specific row of matrix $\mathbf{M}$ indexed by v. The model that emerges from this approach is referred to as Simple-GAINER, abbreviated as SGR. In clear contexts, $\mathbf{s}_v$ is used to represent the particular row indexed by the vertex v in the matrix $\mathbf{A}^{r(v,\tau)}\mathbf{X}^{(0)}$, indicated as $\mathbf{s}_v = [\mathbf{A}^{r(v,\tau)}\mathbf{X}^{(0)}]_v$.

4.4 Significance of the Threshold

We are driven by the intuition of assigning a small value of $r(v, \tau)$ to well-connected nodes, while providing poorly-connected nodes a larger value, thereby extracting the maximum value from the graph structure G. Additionally, we aim for the final smoothed features, $\mathbf{s}_v$, of each node to remain close to the original input features of the node $\mathbf{X}_v^{(0)} = \mathbf{x_v}$, to prevent excessive smoothing. The threshold τ is selected with the precise intention of ensuring that $||\mathbf{s}_v - \mathbf{x}_v||_2$ does not exceed τ, where $|| \cdot ||_2$ represents the l_2 norm.

4.5 Selecting the Node-specific Radius

The value $r(v, \tau)$ is chosen so that $||\mathbf{s}_v - \mathbf{x}_v||_2 \leq \tau$ for all $v \in V$. Mathematically,

$$r(v, \tau) = \min\{l : ||\mathbf{A}^{l}\mathbf{X}^{(0)}]_v - \mathbf{x}_v||_2 \leq \tau\} \tag{4}$$

[1] The acronym GAINER, can also stand for Graph Artificial Intelligence with Node-Exclusive Radius.

The threshold τ, acting as a task-specific hyperparameter, empowers us to meticulously tailor the level of smoothing to meet the task's requirements.

4.6 Neural-GAINER (NGR)

A central query we set out to investigate was whether we could formulate a neural counterpart of SGR, taking into account that GCN acts as the neural counterpart of SGC. One significant obstacle in this formulation is determining how to incorporate layer-specific weight matrices in Equation 1 when nodes possess highly varying radii. Nevertheless, by sharing the same weight matrix, say $\mathbf{W}$, across all nodes and their radii, we introduce a novel GNN architecture known as Neural-GAINER, abbreviated as NGR. The process of NGR on a per-node basis is as follows:

$$\mathbf{X}_v^{(l+1)} = \eta\left([\mathbf{A}\mathbf{X}^{(l)}\mathbf{W}]_v\right),\ l = 0, \cdots, r(v,\tau)-1. \tag{5}$$

The unrolling of Equation 5 allows the GNN to handle information at different radii, similar to the flexibility of recurrent neural networks (RNNs) which enables them to handle variable-size inputs. Unlike an RNN, our NGR aggregates node features from l hops away at every layer l, a unique characteristic that distinguishes the two architectures.

4.7 Computational Complexity Analysis

Let R be the maximum radius $r(v,\tau)$ of all the nodes $v \in V$ of the input graph $G = (V, E)$ and $m = |E|$ be the number of edges in G. The time complexity of the key step of GAINER, i.e., computing node-specific radii is $\mathcal{O}(Rmd)$ where d is the number of input features. The time complexity of training and inference of SGR is $\mathcal{O}(nd^2)$ time where $n = |V|$ is the number of nodes in G and those of NGR is $\mathcal{O}(Rnd^2)$.

5 Experiments

In this section, we empirically validate GAINER's efficacy by conducting extensive experiments including baseline comparison, training time-accuracy tradeoff, memory consumption, sensitivity analyses, etc. The accuracy comparisons are shown in the main text while the other experiments are in the appendix. The tasks considered are

1. Inductive short-text classification,
2. Document classification on attributed graphs,
3. Document coherence assessment.

We have utilised an NVIDIA Titan RTX GPU for training all the models. The training specifics are described in the appropriate subsection dedicated to the given task. Additional details regarding graph construction procedures, datasets, baselines, hyperparameters, and more are given in the appendix following the references.

5.1 Task 1: Inductive Short Text Classification

Short text classification (STC) is a crucial task that has been extensively studied in various NLP applications, including news tagging, efficient information retrieval, sentiment analysis, and query intent classification. In recent times, GNNs have demonstrated remarkable performance in STC by effectively exploiting relevant relational side information through message passing along edges. Recent observations (Zheng et al., 2022; Yang et al., 2021b; Ding et al., 2020) highlight that the majority of models used in this context are transductive models, which lack the ability to handle new texts without undergoing retraining.

5.1.1 Experimental Setup

Progressing towards a more rigourous and practical challenge, we enter the realm of inductive STC, which involves classifying texts that are unseen or unobserved during model training. We adopt the experimental setup of a previous study (Zheng et al., 2022), which addresses inductive short text classification through SimpleSTC by employing a graph structure with words as nodes. We replace the 2-layer GNN on the word graph in SimpleSTC by our GAINER (i.e., SGR, NGR) models.

5.1.2 Model and Training Details

The connection between two words in the word graph is determined by their local co-occurrence statistics, calculated using point-wise mutual information. Our proposed GAINER methods utilise pre-trained word embeddings as node features to smooth and refine the embeddings across the word graph. Short text embeddings are obtained by aggregating node embeddings of the words within the texts, and we predict the class for each short text by training with the cross-entropy loss given by

$$\mathcal{L} = -\sum_{i=1}^{N} (\mathbf{y_i})^{\mathbf{T}} \log(\hat{\mathbf{y}}_\mathbf{i}),$$

where N is the number of training instances, $\mathbf{y_i} \in \{0,1\}^C$ is a one-hot vector of length C that in-

Table 1: Performance Comparison of Different Models on Inductive Short-text Classification.

Dataset →	**Twitter**		**MR**		**Snippets**		**TagMyNews**	
Model ↓	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1
TFIDF+SVM	57.76(1.59)	56.53(1.95)	54.66(0.68)	54.06(0.44)	64.21(1.17)	63.81(0.89)	34.16(1.80)	32.87(1.26)
LDA+SVM	52.71(1.72)	49.08(3.36)	51.86(1.28)	50.98(1.58)	30.16(2.01)	28.71(1.85)	21.45(4.67)	18.19(1.81)
WideMLP	57.60(2.49)	56.51(3.53)	53.12(1.97)	51.41(4.28)	49.55(1.28)	48.69(1.25)	24.79(0.78)	23.97(0.95)
BERT-AVG	50.52(3.61)	47.33(4.17)	50.46(1.68)	48.10(2.95)	66.35(0.46)	65.83(0.88)	62.27(1.61)	56.91(1.00)
BERT-CLS	50.29(0.38)	36.32(4.62)	50.16(0.33)	35.61(1.63)	42.08(10.05)	38.37(10.91)	38.14(5.42)	29.13(4.41)
TLGNN	54.40(3.02)	45.29(8.23)	52.44(1.68)	46.88(7.14)	59.88(2.03)	59.21(2.16)	34.70(1.16)	31.25(1.17)
TextING	61.82(2.19)	60.77(2.44)	58.73(1.02)	58.30(1.26)	76.26(1.20)	75.70(1.41)	60.76(1.35)	57.22(1.27)
HyperGAT	56.12(4.81)	49.92(11.67)	51.59(0.35)	44.81(4.23)	34.91(0.81)	34.80(0.85)	24.43(4.39)	17.77(3.00)
HGAT-inductive	54.88(1.74)	52.51(2.23)	52.21(2.10)	48.48(7.11)	62.56(1.33)	61.98(1.36)	OOM	OOM
SimpleSTC	62.19(1.56)	62.01(1.59)	62.27(1.11)	62.14(1.12)	80.96(1.69)	80.56(2.01)	67.17(1.27)	63.34(1.38)
SimpleSTC-SGC	61.87(1.39)	62.06(1.48)	61.85(0.99)	61.97(1.04)	80.21(1.73)	80.42(1.76)	66.95(1.22)	62.86(1.45)
SGR (Ours)	62.45(1.13)	62.49(1.10)	62.68(0.66)	62.69(0.71)	81.16(1.24)	81.12(1.37)	67.51(0.72)	63.63(0.98)
NGR (Ours)	62.37(1.31)	62.78(1.26)	62.63(0.82)	62.92(0.83)	81.44(1.48)	81.86(1.80)	67.48(1.00)	63.89(1.11)

dicates the true class for instance i, and the predicted class probabilities for instance i across the C classes are contained in $\mathbf{\hat{y}_i} \in [0, 1]^C$.

5.1.3 Experimental Results

In line with previous studies (Zheng et al., 2022; Yang et al., 2021b; Wang et al., 2021a), we remove duplicate texts to ensure fair testing conditions, and then tokenize each sentence while eliminating stop words. To form training and validation sets, we closely follow prior work (Zheng et al., 2022) and randomly pick 20 labeled samples from each class individually. The remaining samples are allocated to the test set, following the same approach as a previous study (Zheng et al., 2022).

The metrics used for comparison are micro-averaged accuracy and macro-averaged F1 score (F1), averaged over five runs on the testing sets, to provide a comprehensive assessment of model performance. We present the experimental findings in Table 1, and for more details on the hyperparameters, description of each baseline, and additional experiments, please refer to the Appendix.

Observations: Based on the table, it is clear that our proposed SGR and NGR methods excel in utilising the word graph structure to its potential, surpassing GNN-based methods with hop size fixed across all nodes.

5.2 Task 2: Document Classification on Text Attributed Graphs

A Text-attributed Graph (TAG) represents a graph structure where nodes correspond to documents, citations between documents serve as edges, and textual attributes such as title and abstract are used to build node features (Yang et al., 2021a; Zhang et al., 2018). The combination of textual attributes and graph topology provides a rich vein of information, enhancing representation learning in important areas such as text classification, recommendation systems, social media analysis, and information retrieval. Recent research has seen a growing interest in integrating language models and GNNs to learn node representations in TAGs (He et al., 2023; Zhao et al., 2023).

5.2.1 Experimental Setup

In our study, we make use of the Cora and PubMed datasets, which were provided with titles and abstracts in a recent study (He et al., 2023). We closely follow the experimental setup of the study including the LM-based pipeline proposed. We replace the 3-layer GCN in the study with our proposed GAINER methods.

5.2.2 Training Details

The node features consist of three distinct components: (i) a fine-tuned language model representation of the text sequence (title and abstract), (ii) a fine-tuned language model representation of the explanation generated by a large language model (LLM), such as ChatGPT, and (iii) the highest-ranked predictions of the document class provided by the LLM (He et al., 2023). The training of our proposed GAINER approaches, SGR and NGR, is performed using the aforementioned node features.

Table 2: Document Classification with LLM features.

Models	Cora	PubMed
GCN	89.35± 0.59	94.31 ± 0.43
SAGE	89.90 ± 1.11	96.18 ± 0.53
GAT	89.39 ± 1.40	96.04 ± 0.47
SGC	89.27 ± 0.82	94.37 ± 0.41
SGR(Ours)	89.48 ± 0.54	96.13 ± 0.39
NGR (Ours)	89.93 ± 1.02	96.21 ± 0.48

Table 3: Document Classification with shallow Bag-of-words as node features. See Section 5.2 for details

Type	Models	Cora	PubMed
Coupled	GCN	81.8±0.5	79.3±0.7
	GAT	83.0±0.7	79.0±0.3
	SAGE	80.7± 0.5	78.0±0.4
	JK-Net	81.8±0.5	78.8±0.7
Decoupled	APPNP	83.3±0.5	80.1±0.2
	AP-GCN	83.4±0.3	79.7±0.3
	PPRGo	82.4±0.2	80.0±0.4
	DAGNN	84.4±0.6	80.9±0.5
Linear	MLP	61.1±0.6	72.7±0.6
	SGC	81.0±0.2	78.9±0.5
	SIGN	82.1±0.3	79.5±0.5
	S^2GC	82.7±0.3	79.9±0.3
Ours	**SGR**	84.1±0.6	81.1±0.6
	NGR	84.6±0.5	81.4±0.4

A cross-entropy loss function is used to train the models $\mathcal{L} = -\sum_{i=1}^{N}(\mathbf{y_i})^{\mathbf{T}}\log(\hat{\mathbf{y}}_\mathbf{i})$.

5.2.3 Experimental Results

In line with the previous study (He et al., 2023), the ratio we used for splitting the datasets was 0.6/0.2/0.2, where 60% of the data was allocated for training, 20% for validation, and 20% for testing. Additionally, we utilised random seeds to ensure the reproducibility of our experiments, enabling the consistent evaluation of our proposed methods on the respective datasets. The metric used for comparison is classification accuracy over 5 different runs with random seeds.

The experimental findings are presented in Table 2. For more details on the hyperparameters, description of each baseline, and additional experiments, please refer to the Appendix.

We have also conducted experiments on the popular Cora and PubMed datasets with bag-of-words node features with commonly used splits (Kipf and Welling, 2017). The results are shown in Table 3.

Observations: Our proposed methods outperform common baselines like GCN, SAGE, GAT, SGC when utilising widely used bag-of-word node features, as shown in Table 3. These results are significant because traditional shallow bag-of-word features are widely used but lack the informativeness of LLM features, as highlighted in Table 2. LLM features provide richer features, emphasising the potential of our approach. We believe GAINER effectively utilises the graph structure, especially when node features offer limited information, making our method particularly valuable.

5.3 Task 3: Document Coherence Assessment

The concept of textual coherence involves creating a sense of flow and logical progression between sentences, ensuring they are not disjointed or randomly ordered, but instead well-connected and organised (McNamara et al., 2010). Coherence plays a pivotal role in determining the quality of a text and has found extensive application in various downstream tasks such as summarisation, dialogue generation, machine translation, and document-level text generation. Recently, graph-based techniques have been developed to connect structurally similar documents, driven by the hypothesis that documents sharing similar connection structures demonstrate comparable levels of coherence.

5.3.1 Experimental Setup

Our approach closely follows the setup of a recent study (Liu et al., 2023), wherein the proposed StructSim models regards sentences and documents as nodes within a graph. The presence of strong semantic relations between nouns in sentences guides the formation of edges, while pre-trained language models are employed to extract node features. We replace the 2-layer GCN in the proposed StructSim model by our SGR and NGR methods.

5.3.2 Model and Training Details

The training corpus is used to construct a graph, which is then employed to train SGR and NGR. During the evaluation phase, new and unseen documents are introduced into the graph, and the model weights are employed to predict the coherence levels of these documents (inductive setting). A cross-entropy loss function is used to train the models $\mathcal{L} = -\sum_{i=1}^{N}(\mathbf{y_i})^{\mathbf{T}}\log(\hat{\mathbf{y}}_\mathbf{i})$.

Model	Yahoo	Clinton	Enron	Yelp	Average
XLNet+DNN	$60.70_{1.03}$	$64.00_{1.36}$	$55.15_{1.14}$	$56.45_{0.94}$	59.10
StructSim	$63.65_{0.74}$	$66.20_{0.81}$	$57.00_{0.81}$	$58.05_{1.21}$	61.23
StructSim-SGC	$63.43_{0.58}$	$66.22_{0.68}$	$56.87_{0.74}$	$58.07_{1.14}$	61.15
SGR (Ours)	$64.38_{0.61}$	$67.05_{0.75}$	$57.47_{0.76}$	$58.63_{1.10}$	61.79
NGR (Ours)	$64.55_{0.76}$	$67.26_{0.69}$	$57.09_{0.73}$	$59.42_{1.17}$	62.18

Table 4: Mean accuracy (std) results on GCDC. Please see Section 5.3 for details.

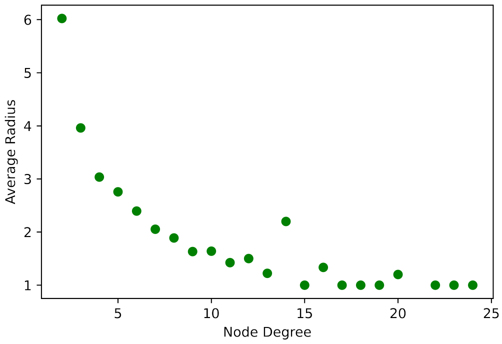


Figure 2: Visualising the relationship between the average node-specific radius of GAINER and the node degrees on the Cora dataset. The plot demonstrates a clear trend: nodes with larger degrees consistently show smaller average radii, whereas nodes with smaller degrees tend to have higher average radii.

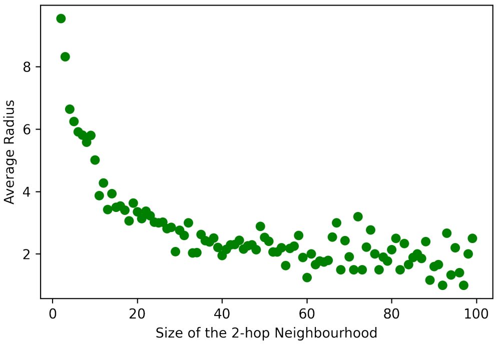


Figure 3: Visualising the relationship between the average node-specific radius of GAINER and the size of the two-hop neighbourhood on Cora. The plot shows a trend that supports our intuition: nodes with good connectivity benefit from smaller radii, and vice versa.

5.3.3 Experimental Results

Table 4 shows the results on the Grammarly Corpus of Discourse Coherence (GCDC) dataset (Liu et al., 2023). We perform 10-fold cross-validation over the training GCDC dataset. Our proposed methods better exploit the structural similarity information between documents, leading to significant improvements compared to recent fixed-hop graph-based approaches, as demonstrated in the table.

Significance: The p-value of a Welch's t-test comparing the accuracy of our proposed models with the accuracy of the most competitive baselines in Tables 1, 3, and 4 is less than 0.001, indicating strong evidence against the null hypothesis.

5.4 Relationship between Node-specific Radius and Node Connectivity

We delve into the fundamental aspect of our proposed methods: the node-specific radius, which serves as the distinguishing feature, enabling them to outperform existing approaches across tasks.

In Figure 2, we examine the relationship between the node degree and the average node-specific radius, averaged across all nodes with a particular degree. The findings depicted in this figure align with Figure 1, indicating that nodes with lower degrees tend to benefit from larger radii, while nodes with higher degrees benefit from smaller radii.

Figure 3 delves into the interplay between the radius and the size of the 2-hop neighbourhood. The number of nodes offers insights into the density of connectivity in the vicinity of a node. The observations corroborate our intuition, indicating that well-connected nodes typically require smaller radii, while nodes with limited connectivity benefit from a larger hops of information propagation.

6 Conclusion

We have introduced GAINER, a novel graph-based learning framework that assigns a dedicated radius to each node, controlling information propagation depth. We propose Simple-GAINER and Neural-GAINER for graph NLP to harness the power of graph structures to advance graph NLP research. Extensive experiments on short text classification, document classification, and coherence assessment demonstrate the significance of GAINER.

Limitations

Our work lays the foundation for various potential extensions and future enhancements.

More Challenging Structures: Our GAINER approach leverages the principle of homophily, which suggests that nodes with similar labels tend to be connected in the graph, a characteristic commonly observed in our target tasks and datasets. In the *heterophilic setting* (Lim et al., 2021; Zhu et al., 2020; Pei et al., 2020), the complexity increases as there are more instances of node pairs with different labels compared to those with the same label, posing a greater challenge for classification or analysis tasks. In the context of *heterogeneous multi-relational graphs*, the inclusion of multiple types of nodes and edges provides an exciting avenue for investigation, offering diverse perspectives and opportunities for exploration. Extending GAINER to handle such settings is an interesting direction to explore.

Multiple Modalities: In the context of expanding the scope of our work, there are several promising directions to explore. Firstly, considering *multi-modal or multi-graph settings* could provide a richer representation of the data by incorporating diverse sources of information such as text, images, or knowledge graphs. This would enable us to capture more comprehensive relationships and dependencies within the data. Additionally, incorporating external knowledge sources, such as ontologies or domain-specific knowledge bases, could enhance the model's understanding and improve its performance on specific tasks.

Transferability: Investigating the *transferability of our methods across different domains or tasks* would be valuable, as it could reveal the generalisability of our approaches and potentially enable knowledge transfer from one domain to another. Transferring the ideas of GAINER to more advancing models such as graph attention (Zhang et al., 2020; Nikolentzos et al., 2020) and sparse structure learning (Piao et al., 2022) is also a potential avenue for further research.

References

Sami Abu-El-Haija, Hesham Mostafa, Marcel Nassar, Valentino Crespi, Greg Ver Steeg, and Aram Galstyan. 2021. Implicit svd for graph representation learning. In *Advances in Neural Information Processing Systems (NeurIPS) 34*, pages 8419–8431. Curran Associates, Inc.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1957–1967.

Aleksandar Bojchevski, Johannes Klicpera, Bryan Perozzi, Amol Kapoor, Martin Blais, Benedek Rózemberczki, Michal Lukasik, and Stephan Günnemann. 2020. Scaling graph neural networks with approximate pagerank. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 2464–2473.

Ming Chen, Zhewei Wei, Bolin Ding, Yaliang Li, Ye Yuan, Xiaoyong Du, and Ji-Rong Wen. 2020. Scalable graph neural networks via bidirectional propagation. In *Advances in Neural Information Processing Systems (NeurIPS) 33*, pages 14556–14566. Curran Associates, Inc.

Xiuying Chen, Mingzhe Li, Shen Gao, Rui Yan, Xin Gao, and Xiangliang Zhang. 2022. Scientific paper extractive summarization enhanced by citation graphs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4053–4062.

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

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (ACL) (Long and Short Papers)*, pages 4171–4186.

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)*, pages 4927–4936.

Hande Dong, Jiawei Chen, Fuli Feng, Xiangnan He, Shuxian Bi, Zhaolin Ding, and Peng Cui. 2021. On the equivalence of decoupled graph convolution network and label propagation. In *Proceedings of The Web Conference (TheWebConf)*, pages 3651–3662.

Fabrizio Frasca, Emanuele Rossi, Davide Eynard, Ben Chamberlain, Michael Bronstein, and Federico Monti. 2020. Sign: Scalable inception graph neural networks.

Lukas Galke and Ansgar Scherp. 2022. Bag-of-words vs. graph vs. sequence in text classification: Questioning the necessity of text-graphs and the surprising strength of a wide MLP. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL) (Volume 1: Long Papers)*, pages 4038–4051.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1263–1272.

Camille Guinaudeau and Michael Strube. 2013. Graph-based local coherence modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL) (Volume 1: Long Papers)*, pages 93–103.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS) 30*, pages 1024–1034. Curran Associates, Inc.

William L. Hamilton. 2020. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159.

Xiaoxin He, Xavier Bresson, Thomas Laurent, and Bryan Hooi. 2023. Explanations as features: Llm-based features for text-attributed graphs.

Keke Huang, Jing Tang, Juncheng Liu, Renchi Yang, and Xiaokui Xiao. 2023. Node-wise diffusion for scalable graph learning. In *Proceedings of the ACM Web Conference (TheWebConf)*, pages 1723–1733.

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)*, pages 3444–3450.

Qian Huang, Horace He, Abhay Singh, Ser-Nam Lim, and Austin Benson. 2021. Combining label propagation and simple models out-performs graph neural networks. In *International Conference on Learning Representations (ICLR)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*.

Thomas N Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

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

Alice Lai and Joel Tetreault. 2018. Discourse coherence in the wild: A dataset, evaluation and methods. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 214–223.

Kwei-Herng Lai, Daochen Zha, Kaixiong Zhou, and Xia Hu. 2020. Policy-gnn: Aggregation optimization for graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 461–471.

Irene Li, Aosong Feng, Dragomir Radev, and Rex Ying. 2023a. HiPool: Modeling long documents using graph neural networks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (ACL)*, pages 161–171.

Juanhui Li, Harry Shomer, Jiayuan Ding, Yiqi Wang, Yao Ma, Neil Shah, Jiliang Tang, and Dawei Yin. 2023b. Are message passing neural networks really helpful for knowledge graph completion? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL)*.

Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser Nam Lim. 2021. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In *Advances in Neural Information Processing Systems (NeurIPS) 34*, pages 20887–20902. Curran Associates, Inc.

Bang Liu and Lingfei Wu. 2022. *Graph Neural Networks: Foundations, Frontiers, and Applications*, chapter 21: Graph Neural Networks in Natural Language Processing. Volume 1 of (Wu et al., 2022).

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 and Data Mining (KDD)*, pages 338–348.

Wei Liu, Xiyan Fu, and Michael Strube. 2023. Modeling structural similarities between documents for coherence assessment with graph convolutional networks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL)*, pages 7792–7808.

Xiaojun Ma, Junshan Wang, Hanyue Chen, and Guojie Song. 2021. Improving graph neural networks with structural adaptive receptive fields. In *Proceedings of The Web Conference (TheWebConf)*, pages 2438–2447.

Yao Ma and Jiliang Tang. 2020. *Deep Learning on Graphs*. Cambridge University Press.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1506–1515.

Danielle S. McNamara, Scott A. Crossley, and Philip M. McCarthy. 2010. Linguistic features of writing quality. *Written Communication*, 27(1):57–86.

Xupeng Miao, Wentao Zhang, Yingxia Shao, Bin Cui, Lei Chen, Ce Zhang, and Jiawei Jiang. 2021. Lasagne: A multi-layer graph convolutional network framework via node-aware deep architecture. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 35(2):1721–1733.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022. Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 4363–4374.

Giannis Nikolentzos, Antoine J.-P. Tixier, and Michalis Vazirgiannis. 2020. Message passing attention networks for document understanding. In *Proceedings of the Thirty-Fourth Conference on Association for the Advancement of Artificial Intelligence (AAAI)*, pages 8544–8551.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-{gcn}: Geometric graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics (ACL).

Yinhua Piao, Sangseon Lee, Dohoon Lee, and Sun Kim. 2022. Sparse structure learning via graph neural networks for inductive document classification. In *Proceedings of the Thirty-Sixth Conference on Association for the Advancement of Artificial Intelligence (AAAI)*, pages 11165–11173.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL)*, pages 101–108.

Yifu Qiu and Shay B. Cohen. 2022. Abstractive summarization guided by latent hierarchical document structure. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5303–5317.

T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. 2023. A survey on oversmoothing in graph neural networks.

Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. 2009. Efficient graphlet kernels for large graph comparison. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 488–495. PMLR.

Indro Spinelli, Simone Scardapane, and Aurelio Uncini. 2021. Adaptive propagation graph convolutional network. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 32(10):4755–4760.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)*, 15(56):1929–1958.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*.

Yaqing Wang, Song Wang, Quanming Yao, and Dejing Dou. 2021a. Hierarchical heterogeneous graph representation learning for short text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3091–3101.

Yifei Wang, Yisen Wang, Jiansheng Yang, and Zhouchen Lin. 2021b. Dissecting the diffusion process in linear graph convolutional networks. In *Advances in Neural Information Processing Systems (NeurIPS) 34*, pages 5758–5769. Curran Associates, Inc.

Yujie Wang, Hu Zhang, Jiye Liang, and Ru Li. 2023. Dynamic heterogeneous-graph reasoning with language models and knowledge representation learning for commonsense question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL)*.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 6861–6871.

Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. 2023. *Graph Neural Networks for Natural Language Processing: A Survey*. Now Foundations and Trends.

Lingfei Wu, Peng Cui, Jian Pei, and Liang Zhao. 2022. *Graph Neural Networks: Foundations, Frontiers, and Applications*. Springer Singapore.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, pages 4–24.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 5453–5462.

Liyan Xu and Jinho Choi. 2022. Modeling task interactions in document-level joint entity and relation extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 5409–5416.

Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. 2021a. Graphformers: Gnn-nested transformers for representation learning on textual graph. In *Advances in Neural Information Processing Systems (NeurIPS) 34*, pages 28798–28810. Curran Associates, Inc.

Tianchi Yang, Linmei Hu, Chuan Shi, Houye Ji, Xiaoli Li, and Liqiang Nie. 2021b. Hgat: Heterogeneous graph attention networks for semi-supervised short text classification. *ACM Trans. Inf. Syst.*, pages 1–29.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5754–5764. Curran Associates, Inc.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022a. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL)*, pages 5773–5784.

Wentao Zhang, Zeang Sheng, Mingyu Yang, Yang Li, Yu Shen, Zhi Yang, and Bin Cui. 2022b. NAFS: A simple yet tough-to-beat baseline for graph representation learning. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pages 26467–26483.

Wentao Zhang, Mingyu Yang, Zeang Sheng, Yang Li, Wen Ouyang, Yangyu Tao, Zhi Yang, and Bin CUI. 2021. Node dependent local smoothing for scalable graph learning. In *Advances in Neural Information Processing Systems (NeurIPS) 34*, pages 20321–20332. Curran Associates, Inc.

Xinyuan Zhang, Yitong Li, Dinghan Shen, and Lawrence Carin. 2018. Diffusion maps for textual network embedding. In *Advances in Neural Information Processing Systems (NeurIPS) 31*, pages 7598–7608. Curran Associates, Inc.

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)*, pages 334–339.

Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2023. Learning on large-scale text-attributed graphs via variational inference. In *International Conference on Learning Representations (ICLR)*.

Kaixin Zheng, Yaqing Wang, Quanming Yao, and Dejing Dou. 2022. Simplified graph learning for inductive short text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10717–10724.

Hao Zhu and Piotr Koniusz. 2021. Simple spectral graph convolution. In *International Conference on Learning Representations (ICLR)*.

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 (NeurIPS) 33*, pages 7793–7804. Curran Associates, Inc.

Appendix GAINER: Graph Machine Learning with Node-Specific Radius

The appendix contains additional details such as dataset statistics, detailed empirical setup, baseline methods used for comparison, the hyperparameter values, and supplementary experiments.

A Task 1: Inductive Short Text Classification

In this section, we describe additional details on the inductive STC problem. We supplement the experiments in the main section with additional experiments such as varying data percentages, embedding sizes, and threshold τ.

A.1 Datasets

This paper has utilised short text datasets from a prior work (Zheng et al., 2022), and we present a summary of the key statistics in Table 5.

Dataset	# texts	l	c	# words
Twitter	9970	6.6	2	20726
MR	10,661	11.2	2	18447
Snippets	10174	17.5	8	25906
TagMyNews	31279	6.5	7	231218

Table 5: Key statistics of short text datasets used, l is the average legnth, and c is the number of classes

A.2 Detailed Empirical Setup

In this subsection, we explain the experimental setup of inductive short text classication in detail. We closely follow the setup of a prior work (Zheng et al., 2022).

A.2.1 Graph Construction

To compensate for the limited availability of semantic information, we initially create a word graph by leveraging WikiText, an extensive external corpus, allowing us to augment the dataset with a broader context and enrich the representation of words. Subsequently, we learn a text graph by learning connections between short texts and the words contained within them. Through this process, we facilitate the propagation of the limited labeled information across the interconnected texts, allowing for the dissemination of valuable insights and enhancing the overall learning process.

A.2.2 Data Preprocessing

Our data preprocessing strategy involves narrowing down the input to solely the abstracts, which encapsulate the key information from each article. Following this, we tokenise the sentences within the abstracts and apply further preprocessing steps, including the removal of stop words and the exclusion of infrequent words that occur less than 10 times in the global pool. By implementing these measures, we curate a refined dataset that prioritises meaningful and frequently occurring content.

A.2.3 Word Graph

To capture the interrelationships between words, we construct a word graph which serves as a representation of the connections among these words. This graph is constructed by establishing connections between words, leveraging local co-occurrence statistics derived from point-wise mutual information calculations.

A.2.4 Model Details

In this step, we generate node embeddings within the word graph, by training our GAINER approaches, i.e., Equation 3 for the SGR model and Equation 5 for the NGR model, to capture both the general topology and the specific characteristics of the dataset. This training process enables us to encode comprehensive representations of the nodes, incorporating both the overall structure of the word graph and the task-specific information required for each STC task. The short texts are encoded as the weighted aggregated node embeddings. The weights are given by term frequency-inverse text frequency (TF-IDF).

A.2.5 Optimisation

In the final step, we predict the class labels for each short text and optimize our model, SimpleSTC, based on the classification loss. This process involves assigning the most appropriate class label to each short text and fine-tuning our model to minimise the cross-entropy classification error.

A.2.6 Inference

During inference, all parameters of GAINER are fixed. We tokenise each short text and obtain its embedding and predict its class.

Note on Word Graph vs. Short Text Graph In this particular configuration, we adopt a hierarchical approach to graph learning that involves two distinct graphs. The first graph with words as nodes is created utilising Point-wise Mutual Information

(PMI), whereas the second graph with short texts as nodes is learned during the training process through the construction of edges based on cosine similarity of trained embeddings. Notably, our focus in this work is primarily on leveraging our proposed GAINER techniques specifically for the word-level graph. However, an intriguing avenue for future investigation involves extending GAINER to hierarchical graph learning and/or incorporate edge learning within the short text graph, which holds potential for further advancements in this domain.

A.3 Baselines

We compare our SGR and NGR methods with:

- Traditional two-step feature extraction and classification methods including TF-IDF+SVM, LDA+SVM (Cortes and Vapnik, 1995), and WideMLP (Galke and Scherp, 2022)

- Pretrained BERT (Devlin et al., 2019) which represents each short text as the averaged word embeddings (BERT-Avg) or the embedding of the CLS token (BERT-CLS) and is fine-tuned together with a linear classifier

- Inductive GNN based text classification methods including TLGNN (Huang et al., 2019), TextING (Zhang et al., 2020), and HyperGAT (Ding et al., 2020), and

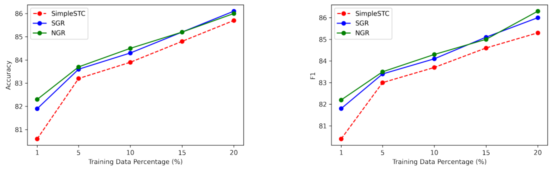
- Inductive STC Methods including HGAT-Inductive (Yang et al., 2021b) and SimpleSTC (Zheng et al., 2022) and SimpleSTC-SGC which is GCN in SimpleSTC replaced by SGC (Wu et al., 2019).

A.4 Hyperparameters

The sliding window size for caclulating PMI is 5 and the word embedding size is 200. We use the Adam optimiser with a learning rate of 0.001 to train for a maximum of 1000 epochs. The dropout rate is 0.9. The threshold for GAINER is selected based on grid search in the range $\tau \in \{0.05, 0.075, 0.1, 0.125, 0.15\}$.

A.5 Effect of Training Data Percentage

Figure 4 illustrates the changes in accuracy and F1 scores on the Snippets dataset as the size of the training dataset varies. The figures vividly highlight the performance gains attained by our proposed methods, particularly in cases where the training dataset size is extremely limited. We attribute this to the enhanced information propagation capabilities of SGR, NGR, allowing them to leverage the rich graph structure more efficiently, especially in scenarios with low supervision.



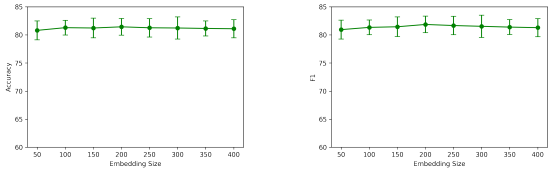
(a) Accuracy

(b) F1

Figure 4: Accuracy and F1 scores of SGR, NGR, and the most competitive baseline (SimpleSTC) with varying training data percentages on the Snippets dataset.

A.6 Effect of Embedding Size

The effect of varying embedding sizes on NGR performance is depicted in Figure 5. The findings suggest that the NGR method is capable of capturing and leveraging meaningful information from the graph structure across a range of embedding sizes. This flexibility in accommodating different embedding sizes enhances the adaptability and robustness of the NGR approach in various applications.



(a) Accuracy

(b) F1

Figure 5: Accuracy and F1 scores of NGR with varying embedding sizes on the Snippets dataset.

A.7 Memory Consumption

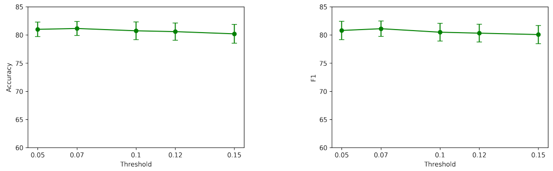
Table 6 shows the memory consumption of SGR, NGR compared to the SimpleSTC. Due to its non-neural nature on the word graph, SGR utilises the least memory. NGR requires the most memory while exhibiting superior overall accuracy.

Model	Twitter	MR	Snippets	TagMyNews
SimpleSTC	9.10	9.20	9.15	12.37
SGR	8.91	9.03	8.97	12.15
NGR	10.01	10.58	10. 53	13.67

Table 6: Memory Consumption of SGR, NGR, and SimpleSTC in GB on different datasets

A.8 Effect of the Threshold

The effect of varying embedding sizes on SGR is depicted in Figure 6. The reason behind choosing SGR for this experiment is its resilience to variations in the threshold. As the node-specific radii tend to increase with decreasing threshold values, this characteristic of SGR does not adversely affect its training process. The findings suggest that the SGR method is capable of capturing and leveraging meaningful information from the graph structure across a range of threshold values. The choice of an optimal threshold value is essential for improving the resilience of models.



(a) Accuracy

(b) F1

Figure 6: Accuracy and F1 scores of SGR with varying threshold τ on the Snippets dataset.

B Task 2: Document Classification on Text Attributed Graphs

In this section, we describe the datasets, experimental setups of the document classification problem in detail.

B.1 Datasets Used

Table 7 summarises the datasets used in the paper. The TAG datasets with LLM features were obtained from a recent study (He et al., 2023). The datasets with bag-of-words features were obtained from a popular work (Kipf and Welling, 2017).

Dataset	#Nodes	#Edges	Task	Metric
Cora	2,708	5,429	7-class classification	Accuracy
PubMed	19,717	44,338	3-class classification	Accuracy
CiteSeer	3,312	4,732	6-class classification	Accuracy

Table 7: Statistics of the TAG datasets

B.2 Experimets on Citeseer

Although the Citeseer dataset lacks titles and abstracts, previous studies (Kipf and Welling, 2017) have explored the dataset using bag-of-words as features and established standard splits. We follow the standard setting and report the experimental results on Table 8. The table highlights the strong performance of our proposed methods when compared to various baselines, which will be discussed in detail in the subsequent subsection.

Table 8: Results on Citeseer.

Type	Models	Citeseer
Coupled	GCN	70.8±0.5
	GAT	72.5±0.7
	JK-Net	70.7±0.7
Decoupled	APPNP	71.8±0.5
	AP-GCN	71.3±0.5
	PPRGo	71.3±0.5
	DAGNN	73.6±0.7
Linear	MLP	61.8±0.8
	SGC	71.3±0.5
	SIGN	72.4±0.8
	S^2GC	73.0±0.2
Ours	**SGR**	73.5±0.5
	NGR	**73.7±0.6**

B.3 Description of Baselines

In this section, we describe the baselines by their main characteristics.

Coupled methods refer to a class of techniques in which the feature propagation and feature transformation steps are tightly coupled within each hidden layer.

- **GCN** (Kipf and Welling, 2017) was initially developed as an efficient convolutional method for semi-supervised classification on graph-structured data, and has now become popular in multiple domains due to its effectiveness and versatility.

- **SAGE** (Hamilton et al., 2017), an inductive framework, utilises node attribute information to effectively generate representations for previously unseen data.

- **GAT** (Veličković et al., 2018) utilises masked self-attention layers to assign distinct weights to nodes within a neighborhood, enabling superior learning of node representations.

- **JK-Net** (Xu et al., 2018), a neural network method, offers flexibility in gathering neighborhood information from different ranges, thereby facilitating a more comprehensive and structure-aware representation.

Decoupled methods refer to a class of methods in which the feature propagation and feature transformation are decoupled.

- **APPNP** (Klicpera et al., 2019) capitalises on the correlation between graph convolution networks (GCN) and PageRank to generate enhanced node representations, leading to improved outcomes.

- **AP-GCN** (Spinelli et al., 2021) employs a halting unit to determine the receptive range of a given node, enabling more adaptive and context-aware information propagation.

- **DAGNN** (Liu et al., 2020) introduces a decoupling approach that separates the representation transformation and propagation steps. This decoupling enables deep graph neural networks to effectively utilize large receptive fields without compromising performance.

- **PPRGo** (Bojchevski et al., 2020) incorporates an efficient page-rank-inspired approximation of information diffusion within graph neural networks (GNNs), resulting in notable speed improvements without sacrificing state-of-the-art prediction performance.

Linear methods, in the context of graph machine learning, pertain to a category of approaches where the feature propagation over the graph follows a linear function of specific graph structural elements, such as the graph Laplacian, the adjacency matrix.

- **SGC** (Wu et al., 2019) simplifies the graph-based learning process by eliminating non-linearities in GCN and collapsing weight matrices between consecutive layers.

- **SIGN** (Frasca et al., 2020) SIGN is a highly efficient and scalable graph embedding method that offers an alternative to graph sampling in GCN. It utilises various local graph operators tailored to different tasks.

- **S^{2}GC** (Zhu and Koniusz, 2021) introduces a modified Markov Diffusion Kernel to create a variant of GCN that balances low-pass and high-pass filtering. This unique approach enables the capturing of both global and local contexts for each node.

B.4 Hyperparameters

The node embedding size of NGR is selected based on grid search in the range $\{32, 64, 128, 256\}$. We use the Adam optimiser with a learning rate of 0.001 to train for a maximum of 1000 epochs. The dropout rate is 0.5. The threshold for GAINER is selected in the range $\tau \in \{0.05, 0.075, 0.1, 0.125, 0.15\}$.

B.5 Training Time, Test Accuracy Tradeoff

In this section, we explore the relationship between training time and test accuracy, examining the trade-off between the two factors. The findings from the PubMed dataset, focusing on the utilization of bag-of-words features, are visually depicted in Figure 7, providing insights into the relationship between training time and test performance. When comparing with linear models such as SGC and S2GC, several notable observations emerge: (a) both coupled and decoupled GNNs demand substantially longer training times, (b) SGR achieves superior test accuracy while maintaining a training time similar to that of SGC, (c) NGR requires more time but also delivers excellent test performance.

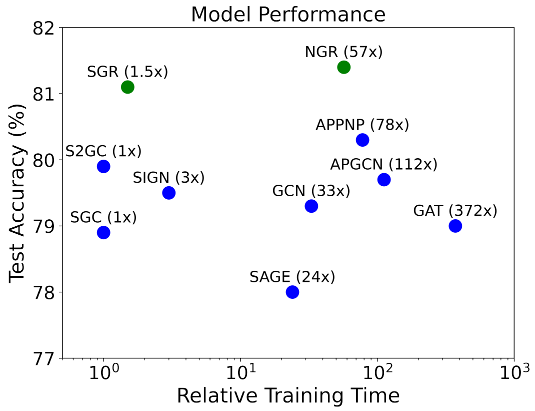


Figure 7: Visualising the relative training times and test accuracy tradeoff of the proposed method (green) and baselines (blue) on the PubMed dataset with bag-of-words features. SGR achieves high test accuracy with impressive speed, while NGR requires more time but also delivers excellent performance.

C Task 3: Document Coherence Assessment

In this section, we delve into the specifics of the document coherence assessment task. In particular, we provide a detailed account of the dataset utilised, the experimental setup employed in our study, comprehensive descriptions of the baseline methods employed, and an overview of the hyperparameters chosen.

C.1 Dataset Used

Our study utilizes the Grammarly Corpus of Discourse Coherence (GCDC) dataset (Lai and Tetreault, 2018) as the benchmark dataset, specifically designed for assessing document coherence. This dataset has recently been used for the task of measuring the coherence of a given text (Liu et al., 2023). The GCDC dataset comprises texts from diverse domains, including *Yahoo* online forum posts, emails from Hillary *Clinton*'s office, Enron *emails*, and *Yelp* online business reviews. Table 9 shows some key statistics of the dataset.

Dataset	Split	#Doc	Avg #W	Max #W	Avg #S
Yahoo	Train	1000	157.2	339	7.8
	Test	200	162.7	314	7.8
Clinton	Train	1000	182.9	346	8.9
	Test	200	186.0	352	8.8
Enron	Train	1000	185.1	353	9.2
	Test	200	191.1	348	9.3
Yelp	Train	1000	178.2	347	10.4
	Test	200	179.1	340	10.1

Table 9: The statistics of the GCDC dataset. #Doc, #W, #S denote the number of documents, words, sentences.

C.2 Detailed Empirical Setup

We closely follow the setup of a recent study (Liu et al., 2023). It consists of four components which we organise as four sub sections

C.2.1 Constructing the Sentence Graph

Our approach to representing a document as a directed sentence graph builds upon a prior work (Guinaudeau and Strube, 2013). However, certain modifications are introduced to enhance the graph construction process. Connections between sentences are established by considering the existence of strong semantic relations between the nouns in those sentences.

To process and segment a document, we employ the Stanza toolkit (Qi et al., 2020) that allows allows us to accurately divide the document into individual sentences and identify all the nouns present in each sentence. To determine the semantic connection between two sentences, we calculate the similarity score (using cosine similarity) for each pair of nouns and selecting on the basis of the maximum similarity score. If the maximum similarity score exceeds a threshold, a directed edge is added between the sentences, resulting in the construction of a directed graph.

C.2.2 Subgraph Set

In this section, we focus on representing sentences through a subgraph set, allowing us to compare graph structures efficiently and enables document comparison based on structure. A subgraph of a graph is such that the nodes in it can be mapped to the nodes in the graph with the same connection relations. We only consider subgraphs without backward edges, as our approach processes documents from left to right.

We use weakly connected and disconnected subgraphs, as they reflect document properties related to coherence. Given a sentence graph, we mine contained k-node subgraphs, filter out distant subgraphs, count their frequency, and identify isomorphic subgraphs to represent the sentence graph as a subgraph set. The aformentioned approach is inspired by a prior study (Shervashidze et al., 2009).

C.2.3 Doc-subgraph Graph

In this section, we introduce the concept of the doc-subgraph graph, which is an undirected graph constructed at the corpus level. It connects structurally similar documents through their shared subgraphs. The graph consists of document nodes and subgraph nodes, with the total number of nodes being the sum of the number of documents and the number of distinct k-node subgraphs mined from the documents.

Two types of edges are defined in the graph: edges between documents and subgraphs, and edges between subgraphs. The first type of edge is determined based on the presence of a subgraph in a document's subgraph set, with the edge weight being a combination of the subgraph's frequency in the set and its inverse document frequency. The second type of edge is constructed between subgraphs that co-occur in the same document's subgraph set, and its weight is calculated using the Pointwise Mutual Information (PMI) measure.

C.2.4 Applying GAINER

The resulting doc-subgraph graph captures the structural relationships between documents and subgraphs, providing a comprehensive representation of the corpus. We apply GAINER methods, viz., SGR and NGR on the aformentioned docsubgraph-graph.

The input to GAINER is the adjacency matrix of the doc-subgraph graph, where self-connections are added to each node. The input node features for the document nodes are representations obtained through a pre-trained language model and zero vectors for the subgraph nodes. The output of GAINER is passed through an activation function and fed into a softmax classifier for prediction.

C.2.5 Training and Evaluation

During training, the model is trained using Cross-Entropy loss over the document nodes, where the labels are one-hot encoded. The doc-subgraph graph is constructed based on the training corpus, and GAINER methods are trained on this graph. During evaluation, the model operates inductively.

For each document in the test corpus, it is added to the doc-subgraph graph, and its adjacency matrix is normalised. The model then predicts the label for the document based on the updated graph. This ensures that the model can make predictions on unseen documents without using information from other samples in the test corpus.

C.3 Baseline Description

The baseline model, XLNet+DNN, utilises document representations obtained from the XLNet model (Yang et al., 2019) as input features. It then learns document embeddings using a two-layer deep neural network (DNN) and employs a softmax layer as the classifier for making predictions. StructSim is the model proposed in the recent study (Liu et al., 2023) which uses GCN on the Doc-subgraph Graph whereas StructSim-SGC uses SGC instead of GCN on the same graph.

C.4 Evaluation Setting and Hyperparameters

To evaluate the performance of our method, we conduct cross-validation experiments on the GCDC dataset and the TOEFL corpus following established practices in the literature. For the GCDC dataset, we perform 10-fold cross-validation on the training dataset, as done in previous work (Lai and Tetreault, 2018). We set the dimensionality of GAINER methods to 240 for the Clinton and Enron domains, and 360 for the Yahoo and Yelp domains. The Adam optimiser (Kingma and Ba, 2015) with an initial learning rate of 0.01 is used for Clinton and Enron, while a learning rate of 0.008 is used for Yahoo and Yelp. Dropout (Srivastava et al., 2014) with a rate of 0.5 is applied, and the model is trained for 160 epochs.