Dissertation on
TEXT CLASSIFICATION USING GRAPH NEURAL NETWORK: A SURVEY

Thesis submitted towards partial fulfilment
of the requirements for the degree of

Master in Multimedia Development

Submitted by
Debolina Roy

EXAMINATION ROLL NO.: M4MMD24002B
UNIVERSITY REGISTRATION NO.: 163783 OF 2022-23

Under the guidance of
Prof. Dr. Matangini Chattopadhyay

School of Education Technology
Jadavpur University

Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata-700032
India

2024

Master in Multimedia Development
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata,India

CERTIFICATE OF RECOMMENDATION

This is to certify that the thesis entitled **"**Text Classification Using Graph Neural Network: A Survey**"** is a bonafide work carried out by Debolina Roy under our supervision and guidance for partial fulfillment of the requirements for the degree of Master in Multimedia Development in School of Education Technology, during the academic session 2022-2024.

SUPERVISOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DIRECTOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DEAN - FISLM
Jadavpur University,
Kolkata-700 032

Master in Multimedia Development
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF APPROVAL

This foregoing thesis is hereby approved as a credible study of an engineering subject carried out and presented in a manner satisfactory to warranty its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not endorse or approve any statement made or opinion expressed or conclusion drawn therein but approve the thesis only for purpose for which it has been submitted.

Committee of final examination for evaluation of Thesis

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this thesis contains literature survey and original research work by the undersigned candidate, as part of her **Master in Multimedia Development** studies.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by this rule and conduct, I have fully cited and referenced all materials and results that are not original to this work.

NAME: **Debolina Roy**

EXAMINATION ROLL NUMBER: **M4MMD24002B**

THESIS TITLE: **Text Classification Using Graph Neural Network: A Survey**

SIGNATURE:
DATE:

ACKNOWLEDGEMENT

I gratefully acknowledge the resourceful guidance, active supervision and constant encouragement of our reverent Professor **Dr. Matangini Chattopadhyay** of School of Education Technology, Jadavpur University, Kolkata, who despite her other commitments could find time to help me in bringing this thesis to its present shape. I do convey my sincere thanks and gratitude to her.

I also thankfully acknowledge my gratefulness to all Professors and staff of the School of Education Technology, Jadavpur University, Kolkata, for extending all facilities to carry out the present study.

I also thankfully acknowledge the assistance and encouragement received from my family members, friends and others during the preparation of this thesis.

…….....

DEBOLINA ROY
ROLL NO -M4MMD24002B
Master in Multimedia Development
JADAVPUR UNIVERSITY
KOLKATA- 700032

CONTENTS

List of Figures

List of Tables

List of Abbreviations

AGGNN: Attention Gated Graph Neural Network
AI: Artificial Intelligence
BiLSTM: Bi Long-Short Term Memory Network
BoF: Bag of Features
BoW: Bag of Words
CLHG: Cross-Lingual Heterogeneous Graph Convolutional Network
CNN: Convolutional Neural Networks
DADGNN: Deep Attention Diffusion Graph Neural Network
DAGNN: Domain Adversarial Graph Neural Network
DRGCN: Document-Relational Graph Convolutional Network
FN: False Negative
FP: False Positive
GAT: Graph Attention Network
GloVe: Global Vectors for Word Representation
GNN: Graph Neural Network
HGAN: Heterogeneous Graph Attention Network
IMDb: Internet Movie DataBase
KNN: K-Nearest Neighbor
LSTM: Long-Short Term Memory Network
ME-GCN: Multi-dimensional Edge enhanced Graph Convolutional Networks
MGNN: Multi Granular Topic Aware Graph Neural Network
MPM: Message Passing Mechanism
MR: Movie Review
NB: Naive Bayes
NLP: Natural Language Processing
PMI: Pointwise Mutual Information
RGNN: Recursive Graph Neural Network
RNN: Recurrent Neural Networks
SST: Stanford Sentiment Treebank
ST-Text-GCN: Self Training Text method based on Graph Convolutional Networks
SVM:Support Vector Machine
TensorGCN: Tensor Graph Convolutional Network
TextGCN: Text Graph Convolutional Network
TextING: Inductive Text Classification
TF-IDF: Term Frequency-Inverse Document Frequency
TLGNN: Text Level Graph Neural Network
TN: True Negative
TP: True Positive
XMTC: Extreme Multi label Text Classification

EXECUTIVE SUMMARY

Text classification is a fundamental challenge in Natural Language Processing (NLP). While many recent models for text classification have relied on sequential deep learning techniques, graph neural networks (GNNs) offer a powerful alternative by directly handling complex structured text data and leveraging global information. Many real world text classification tasks can naturally be framed as graphs, representing words, documents and global corpus features.

In this survey, we studied the methods around previous 25 years research works, with a focus on both corpus-level and document-level GNNs. We explore these methods in depth, discussing their graph construction mechanisms and the learning processes involved in using graph-based approaches. Beyond the technological exploration, we also examine key challenges and future directions in text classification using GNNs.

This survey includes a review of relevant datasets, evaluation metrics and experimental designs, along with a comprehensive comparison of various techniques. We identify the strengths and limitations of different evaluation metrics, offering a clear perspective on their impact on performance. Finally, we present a summary of the state-of-the-art results on publicly available benchmarks, highlighting the progress and gaps in the field.

Chapter 1

1.0 INTRODUCTION

Natural language processing (NLP) is a branch of artificial intelligence (AI) that enables computers to comprehend, generate, and manipulate human language. NLP is a way to extract information from the texts. It helps to know whether the information being shared is genuine or fake. In the field of NLP, text classification is a fundamental problem. There are numerous applications of text classification such as document organization, news filtering, spam detection, opinion mining etc.

An essential part of text classification is text representation. Text representation is very important because text cannot be directly interpreted by a classifier algorithm and needs to be mapped into a vector of numeric weights based on the document's contents. So it directly affects the classification accuracy.

Traditional text classification methods mainly focus on hand-crafted feature engineering. Traditional models accelerate text classification with improved accuracy and make the application scope of traditional expand. Text classification in traditional methods often requires a feature engineering step for text representation. The most commonly used feature is the bag of words feature. Besides, some more complex features have been designed.

Machine learning algorithms often use classifiers such as Support Vector Machine (SVM), Naive Bayes (NB), decision trees, k-means, etc. and they are less time-consuming to train. This text classification method trains in preclassified texts, then builds a specified classifier, and finally classifies texts with unknown class labels. The first thing is to preprocess the raw input text for training traditional models, which generally consist of word segmentation, data cleaning, and statistics. This type of classifier has some disadvantages such as sparse feature vectors, dimensional explosion, and difficult feature extraction. Boolean models, vector space models, probability models, and graph space models are also traditional text classification models.

Then the deep learning method developed rapidly. Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Long-Short Term Memory Network (LSTM) are principal modes for this type of text classification. Compared to traditional text classification models, these models provide better results and achieve significant improvement. These deep learning models can capture semantic and syntactic information in local consecutive word sequences well. But in the case of unstructured

complex data, CNN or RNN are unable to classify the text. Then, a generalised form of CNN, Graph Neural Network (GNN) is used. GNN [9] provides an easy way to do node-level, edge-level and graph-level prediction tasks.

1.1 Problem Statement

A survey on text classification by Graph Neural Network.

1.2 Objectives

- We discuss about the text classification using Graph Neural Network with critical situation of twenty GNN text classification models.
- We discuss about the Graph representation of text.
- We compare the performance of these models on the benchmark dataset.
- We discuss the existing challenges and some potential future work for GNN text classification models.

1.3 Scope

This survey provides a comprehensive overview of GNN. The scope includes,

- Introducing GNNs as powerful tools for modeling relationships in text data presenting common architectures such as static and dynamic graphs and their role in representing relationships within text.
- Reviewing commonly used datasets to assess GNN models' performance in text classification and addressing performance metrics and performance table.

1.4 Organization of Paper

- In chapter 1 we provide basic definition of text classification, uses of text classification, text classification algorithm along with the objective, scope and the organization of this survey.

- In chapter 2 we discuss about GNN, Graph representation of a text, work process of GNN.
- In chapter 3 we discuss about previous research on Text Classification by GNN.
- In chapter 4 we introduce commonly used datasets, performance metrics and performance table.
- In chapter 5 we present conclusion of our survey on GNN, challenges and future work.

1.5 Text Classification

Text classification is a process that classify a document under a predefined category. More formally, if d_i is a document of the entire set of documents *D* and $\{c_1, c_2, ..., c_n\}$ is the set of all the categories, then text classification assigns one category c_j to a document d_i [1]. It helps text analysis by categorising and organising text data. It is a part of text mining [2]. The document of the text classification are two types [2], one is single label and another is multi label. The document belongs to the single class is called single label and the document belongs to the multi class is called multi label. The flowchart of the text classification is shown in Fig. 1.

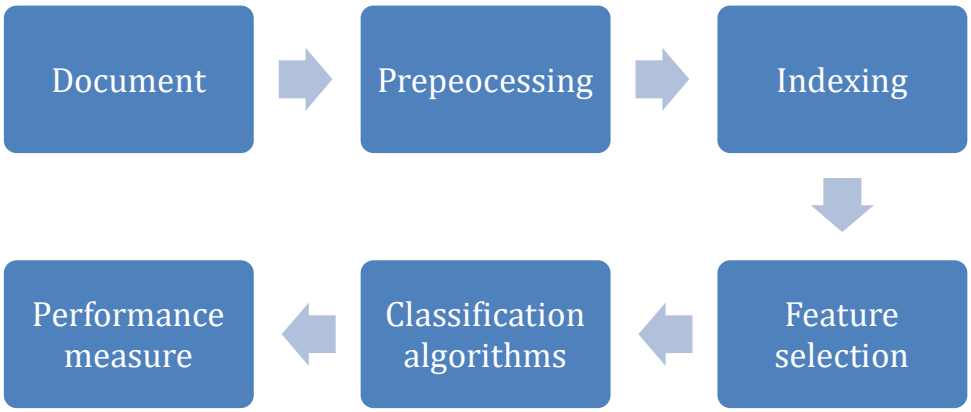

Fig. 1. Classification process.

1.5.1 Document Collection

This is the first step of the text classification. Various type of documents are collected in this step for classification.

1.5.2 Preprocessing

In this step [2] the documents are represented into clear word format. Most of the documents contain many unnecessary words, these words affect the classification. In this step, these words are removed from the document through some processes.

1.5.3 Indexing

After preprocessing, the document text is represented as a document vector. Bag of Words (BoW) and SVM (Support Vector Machine) are the important methods for this. But there are some limitations of high dimensionality, loss of word correlation and the loss of semantic relationship between the term of the document. Term weighting method is used to assign the appropriate weights to the terms. Other various methods are ontology-based representation, N-Grams, Multiword terms, Latent Semantic Indexing etc.

1.5.4 Feature Selection

Feature selection [5, 20] is the fourth stage of the classification. Documents are unstructured data set. This unstructured data must be converted into structured data. Term Frequency-Inverse Document Frequency (TF-IDF) [17], Term Frequency (TF), Word2Vec and Global Vectors for Word Representation (GloVe) are the common techniques of feature extractions.

1.5.5 Classification

The documents are classified in three ways, supervised, unsupervised and semi-supervised methods. The machine learning approaches such as Bayesian classifier, Decision Tree, K-Nearest Neighbor (KNN), Support Vector Machines (SVMs), Neural Networks are the common methods used in this step.

1.5.6 Performance Evaluations

This is the last step of the classification. In this step the performance of the model is calculated. Accuracy is the simplest way of evaluation [2]. It does not work for the unbalanced data set [42].

1.6 Uses of Text Classification

Text classification assigns a predefined label or a class to a given text sequence. This process represents a text document into a numerical representation. Various classifiers are used to predict the categories of the text. Some of algorithms used in text classification are the Naive Bayes family of algorithms, support vector machines (SVM) and deep learning. Application of text classification includes information detection, information retrieval, speech detection, sentiment analysis, topic classification etc.

- Information detection is a task to detect a document whether it is useful or not. Social media is full of abusive language, advertising, fake news etc. So, it is the upmost important job to identify the useful documents in Social media. But there are humongous number of documents in Social media. It is very difficult for human being to identify the correct and most useful documents. So, text classification is the most useful tool for information detection and to save valuable time.

- Information retrieval involves the task of retrieving the relevant theme based on user queries from a text document consisting of huge information. The most important step for document and text data set processing is applying document categorization methods for information retrieval. Text classification enables machine to categorize data and then these data would be retrieved. Naïve Bayes, SVM, decision tree, J48, KNN and IBK methods are used in this field.

- Sentiment Analysis is a process that analyzes text to determine the emotional tone of the text and identify the opinion of the text. A document is classified to determine the opinion to be positive or negative. Naive Bayes and SVM are the methods used for sentiment analysis. Sentiment analysis helps in customer service, business and marketing, social media monitoring, finance and stock market analysis etc.

- Topic classification is a process of classifying a text document into subdomains. A text document is full of information gathered from various sources. So topic classification is a very important application of text classification.

1.7 Text Classification Algorithm

We can categorize text classification algorithms into two types;

a) Traditional machine learning

Traditional machine learning [18, 19] mainly centred on feature engineering and classification algorithm. Bag of Words (BoW) is the most important in feature engineering in NLP, computer vision, spam filter etc. BoW is the simplified representation of the text based on requirement. BoW is a collection of words of a text and semantic relationship between the words is ignored in their construction. In BoW, grammar and word sequence are less important and the number of occurrence of words in the text is counted.

Example of BoW :
Document:
"As the home to UVA's recognized undergraduate and graduate degree programs in systems engineering. In the UVA Department of Systems and Information Engineering, our students are exposed to a wide range of range" [4].
Now the document is tokenized. Then the stopwords and punctuations are removed. Next the words are reduced to their stem and taken one time.

Bag-Of-Word is represented as:
{"home", "UVA", "recognize", "undergraduate", "graduate", "degree", "program", "system", "engineer", "Department", "Information", "student", "exposed", "wide", "range"}.

Bag of Features (BoF)
Feature = {1, 2, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 2}.

This traditional model helps in classification with better accuracy. In this model word segmentation, data cleaning are important preprocessing method of raw data before training. Then, the text is represented as a vector. The individual value of the vector denotes the word frequency corresponding to its inherent position in the text [51]. K-nearest neighbor, support vector machines, decision tree, random forest, naive Bayes, linear

regression, association rules, k-means clustering, are some examples of machine learning techniques.

b) Deep learning

Deep learning plays an important role in today's needs. It can be categorized in to two groups. One type is based on the model of word embedding and another is based on deep neural network. The accuracy of deep learning depends on the effectiveness of word embedding. The word embedding model focuses on word sequence but can't capture contextual semantic information of words. There are various type of deep neural networks with several layers such as input layer, hidden layer, output layer. We can divide deep learning techniques in three types: (i) deep network for supervised learning, (ii) deep network for unsupervised learning and (iii) deep network for hybrid learning.

Chapter 2

2.0 Graph Neural Network

2.1 Definition of Graph

A graph is an abstract collection of data. It is a collection of vertices and edges. A graph is represented as G = (V, E), where V is a set of nodes (vertices) and E is a set of edges of G. A single node in the node set is represented as $v_i \in V$ and $e_{ij} \in E$ denoted the edge between two nodes v_i and v_j respectively. In a graph, two nodes are connected by an edge and it is assumed that every node is connected to itself. Graphs are defined using an adjacency matrix and indicate whether two nodes are connected or not. An adjacency matrix $A \in R^{|V| X |v|}$ can represent the existing edge between nodes. In the case of an unweighted graph, the value associated with that node in the adjacency matrix will be 0 when there is no connection and otherwise 1. The diagonal elements of A are set to 1 because of the self-loop.

2.2 Foundation of GNN

The general mathematical definitions of Graph Neural Networks can be expressed as

$$\mathbf{V^{(l)} = P\ (A,\ V^{(l-1)})} \qquad \mathbf{(1)}$$

where $\boldsymbol{A}$ represents the weighted adjacency matrix and $V^{(l)}$ is the updated node representations at the l-th GNN layers by feeding $(l-1)$-th layer node features $\mathbf{V}^{(l-1)}$ into predefined graph filters P.
The most commonly used graph filtering method is defined as follows:

$$\mathbf{V^{(l)} = Ø(A^{1,} V^{(l-1)}\ W)} \qquad \mathbf{(2)}$$

where $\boldsymbol{A^1 = D^{-1/2}AD^{-1/2}}$ is the normalized symmetric adjacency matrix. $\boldsymbol{A}$ is the adjacent matrix of graph $\boldsymbol{G}$ and $\boldsymbol{D}$ is the degree matrix of $\boldsymbol{A}$, $\boldsymbol{W}$ is the weight matrix and Ø is the activation function. Graph Convolutional Network (GCN) framework for text classification:

$$\mathbf{Y = softmax(\ A^1(ReLU(A^1VW^{(0)})W^{(1)})} \qquad \mathbf{(3)}$$

where $\mathbf{W^{(0)}}$ and $\mathbf{W^{(1)}}$ represent different weight metrics for different GCN layers and $\mathbf{V}$ is the input node features. ***ReLU*** function is used for non-linearization and ***softmax*** is used to generate predicted categories $\boldsymbol{Y}$.

2.3 Preprocessing and Graph representation of Text

Before applying GNN for text classification, a text document must be transformed into a graph after applying preprocessing. This transformation process includes the following five steps,

i. Text cleaning and preprocessing

A text is full of words some of which are unnecessary words such as stopwords, misspelling, slang, etc. We have to remove these words before text classification. It requires some techniques and methods for text cleaning and preprocessing text data.

a. Tokenization

Tokenization is the first step of preprocessing. It is a process that breaks a text into words, phrases and symbols. These parts are called tokens.

Example: “Ram is the best boy in the class”.
The tokens are; ‘Ram’, ‘is’, ‘the’, ‘best’, ‘boy’, ‘in’, ‘the’, ‘class’.

b. Stopwords

Text classification contains many words which don’t play significant role in this process. This type of words are called stopwords such as article, preposition, conjunction, is, are, was, were etc. We have to remove these words in preprocessing.
Example: “Ram is the best boy in the class.”
After removing stopwords, it becomes: “Ram best boy class.”

c. Noise Removal

Most of the text and document data sets contain many unnecessary characters such as punctuation and special characters. These punctuation and special characters are important for human understanding of documents, but these are not important for classification. So, we should remove these noises before classification.

d. Lemmatization

Lemmatization is a process that replaces the suffix of a word with a different one or removes the suffix of a word completely to get the basic word form.

Example: “Falling items conveyor belt.”
After lemmatization it becomes: “Fall item conveyor belt.”

e. Stemming

In a text, one word could appear in different forms while the semantic meaning of each form is the same. Stemming is a method that reduces the words in their base form or ‘stem’.

Example: the words “programming,” “programmer,” and “programs” can all be reduced down to the common word stem “program.”

ii. Graph Construction

We construct a graph for a textual document by representing unique words as vertices and co-occurrences between words as edges. We need to define a new graph structure for a specific task such as designing a word-word or word-document co-occurrence graph. The graph can be classified into many types.

- Corpus-level/Document-level Graph: Corpus-level represents the whole corpus and document-level represents the non-Euclidean relations existing in a single text body.

- Homogeneous Graph/ Heterogeneous Graph: Homogeneous graphs have the same node and edge type and heterogeneous graphs have various nodes and edge types.

- Static Graphs/ Dynamic Graphs: A static graph consists of a fixed sequence of nodes and edges. Node data and edge data are used to represent static graphs. A dynamic graph can be defined as a discrete sequence of static graphs.

- Directed Graphs/ Undirected Graphs: The edges of a directed graph have specified directions and are unidirectional. A directed graph contains a cycle. And for an undirected graph, edges do not have specified direction and it is bidirectional.

iii. Node Construction

Node construction for text classification depends on the node type. According to the node type, there are two types of node construction as discussed in the following.

- Word-level Node Construction:- To numerically represent the node features, many GNN-based text classifications use non-context word embedding methods such as GloVe [43], Word2vec [45], FastText etc. Those embedding methods define the syntactic similarity between words but they are unable to represent complex semantic relationships between the words. The representation of these methods are same and they can't understand the meaning of the out-of-vocabulary words. Recently, there are some studies selecting to get contextual word-level node representation like ELMo, BERT, and GPT. One-hot encoding is the simplest word representation method that is used by a few GNN-based classifiers to achieve state-of-the-art performance.

- Document-level Node Construction:- Document-level node construction is normally performed by aggregating the word-level node construction using some deep learning frameworks. TF-IDF-based document vectors are also used for this type of node construction.

iv. Edge Representation

The edges of a graph are constructed based on the relationship between words and documents. According to the relationship, there are three types of edge representation, defined as follows

- Word-Word Edges- This type of edge is represented by the Euclidean distance between word embedding in each dimension.

- Document-Document Edges- This type is very similar to word-word edges but the documents must share a minimum number of overlapping words.

- Word-Document Edges- Word-Document edges are represented by Term Frequency-Inverse Document Frequency (TF-IDF) values of each dimension.

TF-IDF is represented as the weight of the edges between document-level nodes and word-level nodes.

Pointwise Mutual Information (PMI) measures the co-occurrence between two words in a sliding window W and is calculated as:

$$PMI(i, j) = \log(p(i, j)/p(i)p(j)) \quad (4)$$

$$P(i, j) = W(i, j)/W \quad (5)$$

$$W(i, j) = W(i)/W \quad (6)$$

where, W is the number of windows in total, W(i) and W(i, j) show the number of windows containing word i and both word i and j respectively.

v. Training Setup

This is the last step. GNNs can be divided into supervised, semi-supervised [13] and unsupervised training settings. Supervised training provides labeled training data, while unsupervised training utilizes unlabeled data to train the GNNs. Compared to supervised or unsupervised learning, semi-supervised learning methods are broadly used by GNNs designed for text classification applications. The learning method can be classified into two types:

a) **Inductive Learning:** In this learning, the model collects knowledge from particular examples and then generalizes it so that it predicts outcomes from new data [6]. Decision trees, k-nearest neighbours and neural networks are the algorithms used in this method.
b) **Transductive Learning:** In this learning [7], the model uses specific examples to predict other specific examples without generalizing. This model focuses on making predictions specifically for the given dataset. Prediction of dataset is more accurate.

Node-level and Graph-level tasks involve node or graph classification, clustering, regression, etc. and Edge-level tasks include link prediction or edge classification for predicting the existence of relation between two nodes or the corresponding edge categories.

2.4 Work Process of GNN

In the traditional algorithms, there are some limitations. To overcome the limitations of traditional graph-based algorithms and to represent the non-

Euclidean relations better, Graph Neural Networks are proposed. It simply performs node-level, edge-level and graph-level prediction tasks. To train GNN, there are two main steps described as follows:

a) Message Passing

At each layer of GNN, this step involves aggregating information from neighbouring nodes. For node v at layer k, the aggregated message m^k_v is calculated as a function of the embedding of its neighbours N(v):

$$\mathbf{m^k_v = M^k(h^{k-1}_u, h^{k-1}_v, e_{uv})} \qquad \mathbf{(7)}$$

where, M^k =Message aggregate function at layer k, h^{k-1}_u and h^{k-1}_v are the embeddings of node u and v in the previous layer and e_{uv} represents any edge-specific attributes.

b) Node update

After aggregating messages, the nodes are updated. This is done through a function U^k at layer k. It is given by

$$\mathbf{h^k_v = U^k(h^{k-1}_v, m^k_v)} \qquad \mathbf{(8)}$$

Graph Convolutional Network (GCN) is a simple type of GNN that can capture high-order neighbour node information. The goal of GCN is to implement mapping nodes to a d-dimensional embedding space so that similar nodes in the graph are embedded close to each other.

Chapter 3

3.0 Literature Survey

Jingyu Wang [22] explores the application of GNN for text classification using the Cora dataset. Unlike traditional CNNs, GNNs can extract detailed features from graph-like data structures. This paper emphasizes the significance of preprocessing raw data into graphs to enhance the accuracy and efficiency of GNNs. The core idea is to represent texts and words as vertices in a graph, with edges denoting relationships based on word co-occurrence and frequency. A two-layer GNN model is used, incorporating the ReLU activation function, to classify text data effectively.

Zhang et al. [23] proposed a GNN-based method for Inductive Text Classification (TextING). TextING constructs individual graphs for each document and employs GNNs for learning of word representations based on local structures and generates embeddings for unseen words. The final document embedding is obtained by aggregating the word nodes. Unlike global structure methods, TextING focuses on detailed word interactions within documents, enabling better generalization to new words and inductive learning. To enhance the performance, a multi-channel variant, TextING-M is introduced. TextING comprises three key components- the graph construction, the graph-based word interaction and the readout function.

Contributions of TextING include (a) a graph-based method that constructs individual graphs for each document to learn contextual word interactions, (b) effective inductive learning, making it suitable for scenarios with new, unseen words and (c) superior performance over traditional and graph-based text classification methods, validated through rigorous experiments on multiple datasets.

The experiment highlights TextING's robustness in handling inductive conditions, particularly when training data is limited and many test words are unseen. This is reflected in significant performance gains compared to baselines under such conditions. The model's sensitivity to parameters like interaction steps and graph density was also explored, revealing the optimal setting for best performance.

Deng et al. [24] proposed Attention Gated Graph Neural Network (AGGNN) for updating semantic information of each node from their attention-weighted local neighbors. AGGNN can capture the semantic relationship between the words and every document is represented as a text graph. An attention-based pooling layer, TextPool, is implemented by global and local attention mechanisms to extract the most significant word

nodes as components of document embedding, as the category of text is usually determined by several keywords. In order to make the pooling strategy adapt to documents of different lengths and improve the accuracy of text classification, global and local attention mechanisms are applied. AGGNN comprises three gates; attention gate, update gate and reset gate.

The attention score is calculated by,

$$\mathbf{att_i^t = softmax(h_i^{t-1}) = \frac{exp(h_i^{t-1}w)}{\sum_{j\in N} exp(h_j^{t-1}w)}} \quad \textbf{(9)}$$

where w denotes a shared linear transformation, N denotes the set of 1-hop neighbors of center node and the softmax function is employed to normalize the attention vector att_i^t.

Global attention and local attention are calculated respectively as follows,

$$\mathbf{F_i(h^t) = softnax_i(h^t w) = \frac{exp(h_i^t w)}{\sum_{j\in N} exp(h_j^t w)}} \quad \textbf{(10)}$$

$$\mathbf{F_i(h^t) = \{\frac{\hat{A}_{ii} exp(h_i^t w.\theta)}{\sum_{j\in N}(\hat{A}_{ij} exp(h_j^t w.\theta))}\} \sum_{j\in N} \hat{A}_{ij}} \quad \textbf{(11)}$$

where h_i^{t-1} is the word node and its 1-hop neighbour nodes are h_j^{t-1} , w is a linear transformation applied in all nodes and θ is a parameter and $\hat{A}_{ij}$ denotes the distance between the node i and j.

Yao et al. [25] build a corpus-level graph based on relations of document and word and use Text Graph Convolutional Network (TextGCN) for the corpus.TextGCN is initialized with the one-hot representation of the document and word. TextGCN also learns predictive word and document embeddings. A two-layer GCN performs better than a layer GCN but more layered GCN can't perform as well as two-layer GCN. The weight of the edge between a document node and a word node is the TF-IDF of the word in the document and PMI is used to calculate weight between two word nodes. The semantic co-relationship of words depends on the PMI values. A positive PMI value implies the semantic co-relationship of words in a corpus while a negative PMI value means there is little or no semantic co-relationship of words in a corpus. So, only positive PMI values are used for word pairs. The two-layer GCN allows the information exchange between pairs of documents. The forward movement is given by,

$$\mathbf{Z = softmax(\tilde{A}ReLU(\tilde{A}XW_0)W_1)} \quad \textbf{(12)}$$

where $\widetilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}, \mathbf{softmax}(\mathbf{x_i}) = \frac{1}{\mathbf{z}}\mathbf{exp}(\mathbf{x_i})$ and $\mathbf{z} = \sum_{\mathbf{i}} \mathbf{exp}(\mathbf{x_i})$.

The loss function is defined as;

$$\mathbf{L} = -\sum_{\mathbf{d}\in\mathbf{Y_D}} \sum_{\mathbf{f=1}}^{\mathbf{F}} \mathbf{Y_{df}} \mathbf{ln} \mathbf{Z_{df}} \qquad \textbf{(13)}$$

where, Y_D is the set of labelled document indices, F is the dimension of the output features and Y is the label indicator matrix.

Xien Liu et al. [26] proposed Tensor Graph Convolutional Network (TensorGCN) to describe semantic, syntactic and sequential contextual information. In the text graph tensor, two types of propagation learning are performed, these are intra-graph propagation and inter-graph propagation. To aggregate information from neighbourhood nodes in a single graph, intra-graph propagation is used and to combine heterogeneous information between graphs inter-graph propagation is used. TensorGCN is very effective in uniting various information from different kinds of graphs. The edge of the graph is based on semantic information, syntactic dependency, and local sequential context. To construct a semantic-based graph from text documents, a LSTM-based method is used. To obtain word semantic features cosine similarity between words is calculated. If the similarity value exceeds a predefined threshold then it means that the two words have a semantic relationship in the current document. The edge weight of each pair of words can be defined by,

$$\mathbf{d_{sem}}(\mathbf{w_i}, \mathbf{w_j}) = \mathbf{N_{sem}}(\mathbf{w_i}, \mathbf{w_j}) / \mathbf{N_{total}}(\mathbf{w_i}, \mathbf{w_j}) \qquad \textbf{(14)}$$

where d_{sem} denotes the edge weight between words w_i and w_j, N_{sem} is the number of times that the two words have semantic relationship over all documents in the corpus ssand N_{total} is the number of times that the two words exist in the same document over the whole corpus.
Stanford CoreNLP parser is used to extract the syntactic dependency between words for each document in the corpus. The number of times for each pair of words having syntactic dependency over the whole corpus is calculated. The edge weight is calculated as,

$$\mathbf{d_{syn}}(\mathbf{w_i}, \mathbf{w_j}) = \mathbf{N_{syn}}(\mathbf{w_i}, \mathbf{w_j}) / \mathbf{N_{total}}(\mathbf{w_i}, \mathbf{w_j}) \qquad \textbf{(15)}$$

where d_{syn} denotes the edge weight between words w_i and w_j , N_{syn} is the number of times that the two words have syntactic dependency relationship

over all documents in the corpus and N_{total} is the number of times that the two words exist in the same document over the whole corpus.

Sequential context defines the local co-occurrence between words and PMI is used to define this relation. The edge weight is defined as,

$$\mathbf{d_{seq}(w_i, w_j) = \log(p(w_i, w_j)/p(w_i)p(w_j))} \qquad \mathbf{(16)}$$

where $p(w_i, w_j)$ denotes the probability of the word pair (w_i, w_j) co-occurring in the same sliding window, $p(w_i)$ and $p(w_j)$ are the probability that the word w_i and w_j are occurring in a given window over text corpus respectively.

Chongyi Liu et al. [27] proposed a Document-Relational Graph Convolutional Network (DRGCN) where cumulative TF-IDF document-document relation is used as a feature. This method gives the highest accuracy and the documents and words are represented as the node of the graph. Document relations are calculated as follows:

$$\mathbf{A_{ij} = \log(A(row_i) \cdot A(column_j))} \qquad \mathbf{(17)}$$

where $\mathbf{A(row_i)}$ is the row vector for document **i**, and $\mathbf{A(column_j)}$ is the column vector for document **j**.

The word-document relation is the **TF-IDF** value from the following equation,

$$\mathbf{A'_{ij} = \log(A(row_i).A(column_j))} \qquad \mathbf{(18)}$$

The word-word relation is the PMI value calculated below,

$$\mathbf{PMI(i, j) = \log(p(i,j)/p(i)p(j))}$$
$$\mathbf{P(i, j) = W(i, j)/W}$$
$$\mathbf{P(i) = W(i)/W} \qquad \mathbf{(19)}$$

The total adjacent matrix can be defined as follows:

$$\mathbf{A'_{ij} = \log(A(row_i).A(column_j)) \quad where\ i.j\ are\ documents}$$
$$\mathbf{A'_{ij} = PMI(i,j) \quad where\ i,j\ are\ word}$$
$$\mathbf{A'_{ij} = TF - IDF_{ij} \quad where\ i\ is\ document, j\ is\ word}$$
$$\mathbf{A'_{ij} = 1 \quad where\ i = j}$$
$$\mathbf{A'_{ij} = 0 \quad otherwise} \qquad \mathbf{(20)}$$

Sun et al. [28] proposed a Heterogeneous Graph Attention Network (HGAN) model. Any missing adjacency knowledge can affect the classification. This model extracts the information from heterogeneous graphs and maintains consistency in the result. This model introduces two types of attention mechanisms. These are type-level attention and node-level attention.

The type-level attention is calculated as,

$$\mathbf{a_t = softmax(\sigma(\mu_t.[h_i||h_t]))} \quad \mathbf{(21)}$$

where $\boldsymbol{\sigma}$ is a **ReLU** activation, $\boldsymbol{\mu_t}$ denotes the attention of the type t of the node, operation || is a concatenation, $\mathbf{h_i}$ and $\mathbf{h_t}$ are a specific node and type embedding.

Wei et al. [29] proposed a model that uses recurrent structure to capture contextual information at the time of word representation. In this model, a max pooling layer is used to automatically find out the keyword for text classification. It keeps word order better than GNN but uses the idea of message passing and node update as GNN. This model is divided into three phases- context representation learning, word representation learning, and text representation learning.

In context representation learning BiLSTM is used as the recurrent structure to generate left-side and right-side contexts representations. BiLSTM process is the input in the forward left-to-right and the backward right-to-left directions.

In word representation learning, a node aggregates the most important features from their neighbours and update the features as their own features. In this phase, less important information are ignored.In text representation learning, for pooling operation all nodes are used and a graph is produced. Using this pooling layer, all the information of the text are captured.

Huang et al. [30] proposed Text Level Graph Neural Network (TLGNN) that create graphs for each text. It doesn't capture the dependency of each text and whole corpus. It consumes less memory. To construct the graph, all the words of the text are considered as node and edges connect two adjacent words. The graph is designed as follows,

$$\mathbf{N = \{r_i | i \in [1, l]\}} \quad \mathbf{E = \{e_{ij} | i \in [1, l]; j \in [i - p, i + p]\}} \quad \mathbf{(22)}$$

where, N and E are the node set and edge set of the graph. $\mathbf{r_i}$ denotes the i^{th} word and p denotes the number of adjacent words connected to each word in the graph. This process reduces number of nodes and edges of the graph.

After constructing the graph a non-spectral method Message Passing Mechanism (MPM) is used for convolution. MPM collects information from the adjacent nodes and update representation based on its original and collected information, defined as follows,

$$\mathbf{M_n = max_{a \in N_n^p} e_{an} r_a r'_n = (1 - t_n) M_n + t_n r_n} \qquad \textbf{(23)}$$

where, **Mn** denotes the messages that node n receives from its neighbors. max is a reduction function which combines the maximum values on each dimension to form a new vector as an output, N_n^p denotes the nodes that represent the nearest p words of n in the original text. e_{an} is the edge weight from node a to node n and r_n denotes the former representation of node n. t_n is a trainable variable for node n that indicates how much information of r_n should be kept and r'_n represents the updated n node.

MPM represents the nodes by using the information from the text. The parameters of the graph are used from the global shared metrics. The label of the text is predicted as follows,

$$\mathbf{y_i = softmax(ReLU(W \sum_{n \in N_i} r'_n + b))} \qquad \textbf{(24)}$$

where **W** is a matrix that mapping the vector into an output space, $\mathbf{N_i}$ is the node set of text **i** and **b** is bias.

Meng et al. [31] proposed a Multi-layer Convolutional Neural Network based on prior graph knowledge. This model enhances the semantic information of the text and reduces the dependency of the model on large-scale samples. It also reduces the over-fitting problem. This model consists of three convolutional layers with different size of convolution windows and three pooling layers. The input to this model is in the form of sentence matrix and words are initialized randomly by using any trained word vector. From the various length of sentences, a uniform length is set up by padding operation. The input matrix of the model is given as follows,

$$\mathbf{S = [S_1, S_2 \ldots\ldots S_n]} \qquad \textbf{(25)}$$

The input matrix must contain the prior knowledge and the uniform length can be changed according to this prior knowledge. Each convolutional

layer consists of convolutional kernel. The local features generated by the filter of the word $s_i : s_{i+1}$ is given by,

$$\mathbf{x_i = f(w * S_i; S_{i+1} = b)} \qquad \mathbf{(26)}$$

where, b is an offset term and f is a nonlinear function.

To reduce the number of parameter and minimize the over-fitting problem maximum pooling is used. These maximum values are used for training. Pooling windows of different sizes are used for maximum features. Feature vector is input in the softmax layer for classification. Batch Normalization layer is added to optimize the model between fully connected layer and softmax layer. Current loss function is very important to improve the accuracy.

Zong et al. [32] proposed GNN-XML model for Extreme Multi label Text Classification (XMTC). In this model first a label graph is constructed by the corelation matrix and then attributed graph is built by performing graph convolution with a low-pass graph filter to model label dependencies and label feature. A keyword co-occurrence graph is constructed first where nodes of the graph are keywords. If two nodes are present in one sentence then there is an edge between them and the weight of the edge is number of sentences in which they present.

Li et al. [33] proposed Recursive Graph Neural Network (RGNN) to solve the over smoothing problem. This model helps to represent a text in the form of a graph. In this model LSTM is used. It helps to decide which part of the aggregated neighbour information should be transmitted to upper layers. This is the process of alleviating problem. A global graph-level node is designed to exchange the global and local information. Various type of single and multi label text classification are experimented using this model.

Chong Zhang et al. [34] proposed a type of text classification using tree based graph neural network. In this model, there are dependency matrix for each text and the structural entropy is minimized. According to the structure of this tree, the representation of the non-leaf nodes are updated in layer by layer. A graph coding algorithm is designed to minimize the entropy and to transform the graph into the tree.

Cui et al. [35] proposed a Self Training Text method based on Graph Convolutional Networks (ST-Text-GCN). For semi-supervised short text classification, the number of labeled data is limited and the most of the

models focus on the generation of the text samples which is not reliable. But ST-Text-GCN measures the confidence of each word to calculate the degree of ambiguity of each word. This confidence also helps to calculate the edge weight of the graph to reduce the error. Here, the text graph is undirected. The word confidence depends on labeled documents with predictive labels and calculated as follows:

$$\mathbf{Con_{wi}} = \begin{cases} \frac{\mathbf{Max(C_0, C_1, \ldots\ldots, C_k}}{\sum_{\mathbf{C=0}}^{\mathbf{k}} \mathbf{C_c}} & \textbf{Word in labeled document} \\ \frac{\mathbf{1}}{\mathbf{k}} & \textbf{Others} \end{cases} \quad \textbf{(27)}$$

The adjacency matrix is calculated as follows:

$$\mathbf{A_{m,n}} = \begin{cases} \mathbf{PMI(m.n)} & \mathbf{m, n\ are\ words, PMI(m, n) > 0} \\ \mathbf{TF - IDF_{mn} * Con_m} & \mathbf{m\ is\ the\ word, n\ is\ the\ document} \\ \mathbf{0} & \textbf{Others} \end{cases} \quad \textbf{(28)}$$

The words with higher confidence than threshold are used. Here, two layer GCN is used with one hot vector.

Kunze Wang et al. [36] introduced Multi-dimensional Edge enhanced Graph Convolutional Networks (ME-GCN). This model enhanced semi-supervised text classification by leveraging multi-dimensional edge features in GCNs. Traditional GCN primarily utilizes single-dimensional edge features which limits their ability to capture complex relationships within text data. In this model, the edges of the graph are represented by TF-IDF values. Firstly Word2vec and Doc2vec embedding are trained on the given corpus. The trained embedding also serves as the input embedding of the graph nodes.

Yonghao Liu et al. [37] proposed Deep Attention Diffusion Graph Neural Network (DADGNN). This model helps in text representation and relate a word with its distant neighbours. In this model, a text graph is constructed and each node is initialized a d-dimensional word embedding vector. An edge starts from a target node and ends at its p-hop adjacent nodes. So, the graph becomes directed and the transition matrix is symmetric. The forward propagation formula is given by,

$$\mathbf{H^{(l+1)}} = \boldsymbol{\sigma}(\breve{\mathbf{A}}(\ldots(\boldsymbol{\sigma}(\breve{\mathbf{A}}\mathbf{X}\mathbf{W^{(0)}})\ldots)\mathbf{W^{(l-1)}}) \quad \textbf{(29)}$$

Then, the normalized attention weights between directly connected nodes are calculated. The graph attention diffusion matrix, T is obtained based on the attention matrix **A** as follows,

$$\mathbf{T} = \sum_{\mathbf{n=0}}^{\infty} \boldsymbol{\zeta}_{\mathbf{n}} \mathbf{A}^{\mathbf{n}} \qquad \textbf{(30)}$$

Then, the attention coefficients are obtained by creating attentional links between the unconnected nodes. First, the attention diffusion for each node k are computed independently and then aggregate them. Then, the final representation of all nodes are calculated.

Ziyun Wang et al.[38] proposed Cross-Lingual Heterogeneous GCN (CLHG) to incorporate heterogeneous information between languages. This method aims to incorporate both semantic and syntactic information through the construction of a heterogeneous graph. By treating documents and words as nodes and linking them with various relationships such as part-of-speech roles, semantic similarity and document translations. CLHG focuses on enriching the classification process. This approach not only utilizes the benefits of GNN but also addresses the limitations of existing method. At first, a heterogeneous graph is constructed based on the text information and then all the nodes are encoded with multilingual pretrained language models. Next these nodes are passed to the heterogeneous GCN. The unlabeled data are included in domain documents from the target language. In this method, two types of edges are defined in the graph. For the document-word edges, the POS tags are added to the edges and doc-doc edges, the nodes are connected with their original text. The nodes are encoded first and the feature of these nodes are fixed. In this model, traditional GCN is applied in heterogeneous graph. A linear transformation is applied to get predicted value.

Man Wu et al.[39] proposed Domain Adversarial Graph Neural Network (DAGNN) to overcome the limitations of cross-domain text classification. To capture non-consecutive and long distance semantics, DAGNN uses a graph to model each document at instance level and in feature level, it uses graphs from different domains and jointly train them. At the learning level, domain-adversarial method optimally classifies the text and separate the domain. In this model, the graph is undirected and the content words are the vertices of the graph. PMI is used to calculate the weight of the edges of the graph. Then based on the graph representation, a hierarchial pooling network is developed that employs a GCN to understand the representation in document level. Finally, a domain-adversarial learning is applied for cross domain classification.

Yongchun Gu et al.[40] proposed Multi Granular Topic Aware Graph Neural Network (MGNN) for text classification. Multi-granularity relations are established on a text graph for the triple node set of word,

document and topic. The text graph is constructed based on the word-cooccurrence and document-word relations. Then, a multi-granular topic aware graph is constructed to introduce topic information to the document. There are three types relations between information. First is upper level information (abstract information), second is middle level information (ego information of document) and the last one is underlying information (the integrated information over all words).

Haitao Wang and Fangbing Li [41] propose a text classification model based on Long Short Term Memory network (LSTM) and Graph Attention Network (GAT). In this model, separate graphs are constructed based on syntactic structure of each document.Using LSTM, this model generates word embedding with information of the text and using GAT learns inductive representation of documents. This model consists of three modules: syntax module, LSTM module, GAT module. In syntax module, input text is converted to text graph and the graph is undirected to extract the feature better. The graph is represented by adjacency matrix. This model reduces the number of nodes and edges of the graph than other modules. LSTM is a variation of RNN with several neural units. GAT module is used to pass and update information between nodes. It assigns different weights to different number of nodes. A multi-headed attention mechanism is used for learning process and K independent attention mechanisms is used to calculate hidden nodes. To enrich the feature representation of the node, two layer GAT is used. Softmax is used in this model to predict the labels.

Chapter 4

4.0 Evaluation Method and Analysis

4.1 Datasets

There are many datasets for text classification. In this work, we consider datasets based on GNN. Some widely used benchmark datasets including R8, R52, Movie Review (MR), Ohsumed, 20 NG, Twitter, AGNews, Stanford Sentiment Treebank (SST), Internet Movie DataBase (IMDb), Yelp 2014 etc. are used here.

R8 and R52 are two subsets of Reuters 21587 dataset. These contain 8 and 20 classes associated with single topic respectively. There are 5485 training documents, 2189 test documents and 7688 vocabulary items in R8. There are 6532 training documents, 2568 test documents and 8892 vocabulary items in R52 dataset.

MR, a movie review dataset [50] for binary sentiment classification has 5331 positive as well as 5331 negative reviews.

Ohsumed [46] is a subset of medical dataset, MEDLINE 10. Diseases are divided into 23 categories. The 7,400 documents are used.

20 NG [47] is a widely used dataset of 20 newsgroup. There are a total of 18846 documents and 42757 vocabulary item where 11314 documents are used for training and 7532 for testing.

Table 1. Commonly used in GNN based Text Classification dataset

Name	Domain	Cat.	Docs	Train	Test	Words	Ave. Len.	Models
R8	News	8	7674	5485	2189	7688	65.72	23,24,25,26,27,28,29,30,33,34,36,37,40
R52	News	52	9100	6532	2568	8892	69.82	23,24,25,26,27,29,30,33,34,36,37,40
MR	Movie review	2	10662	7108	3554	18764	20.39	23,24,25,26,27,28,29,33,34,35,36,37,40
Ohsumed	Bibliography	23	7400	3357	4043	14157	135.82	23,24,25,26,27,28,30,33,34,35,36
20 NG	News	20	18846	11314	7532	42757	221.26	25,26,27,29,36
IMDb	Movie review	2	50000	25000	25000	71278	232.77	31,37

Name	Domain	Cat.	Docs	Train	Test	Words	Ave. Len.	Models
AG-News	News	4	127600	120000	7600	128515	44.03	28,31,35,36,37
DBLP	Bibliography	6	81479	61479	20000	25549	8.51	37
DBpedia	Wikipedia	14	630000	560000	70000	-	-	31
SST-1	Movie review	5	11855	9465	2210	19524	20.17	33,37
SST-2	Movie review	2	9613	7792	1821	17539	19.67	37
Twitter	Twitter	2	10000	-	-	-	-	28

4.2 Performance Metrics

To evaluate and compare the performance of proposed models, accuracy and F1 are the most commonly used metrics. True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN) represent the number of true positive, false positive, true negative and false negative samples. N is the total number of samples.

- **Accuracy and Error Rate**: These are the basic evaluation metrics [48, 49] adopted by many GNN-based text classifiers.

$$\textbf{Accuracy} = (TF + TN)/ N, \quad (31)$$

$$\textbf{Error Rate} = 1- \text{Accuracy},$$

$$= (FP + FN)/ N, \quad (32)$$

- **Precision, Recall and F1**: Precision, recall and F1 are used to measure the performance of unlabelled datasets. Precision is used to measure the result relevancy and recall helps to measure the number of truly relevant results acquired. F1 is the harmonic mean of precision and recall. It is calculated from Precision and Recall. These are defined as follows,

$$\text{Precision} = TP/(TP + FP), \quad (33)$$

$$\text{Recall} = TP/(TP + FN), \quad (34)$$

$$F1= (2* \text{Precision} * \text{Recall})/(\text{Precision} + \text{Recall}), \quad (35)$$

There are three types of F1 as follows,

- **Macro-F1:** It applies the same weights to all categories. Macro-F1 is calculated by using the arithmetic mean. It is calculated as;

$$F1_{macro} = \frac{1}{C}\sum_{i=1}^{C} F1_i \qquad (36)$$

- **Micro-F1:** It is calculated by using overall P_{micro} and R_{micro} as follows;

$$F1_{micro} = \frac{2xP_{micro}xR_{micro}}{(P_{micro}+R_{micro})} \qquad (37)$$

Where , $P_{micro} = \frac{\sum_{i\in C} TP_i}{\sum_{i\in C} TP_i+FP_i}$, $R_{micro} = \frac{\sum_{i\in C} TP_i}{\sum_{i\in C} TP_i+FN_i}$ **(38)**

- **Weighted-F1:** It is the weighted mean of F1 of each category where the weight W_i is related to the number of occurrences of the corresponding i th class, which can be defined as:

$$F1_{macro} = \sum_{i=1}^{C} F1_i x W_i \qquad (39)$$

4.3 Performance Table

Table 2. Test Accuracy Comparison of Different Models.

Method	R8	R52	20NG	Ohsumed	MR	IMDb
TextING	98.04±0.25	95.48±0.19	-	70.42 ± 0.39	79.82±0.20	-
TextING-M	98.13±0.12	95.68±0.35	-	70.84 ± 0.52	80.19±0.31	-
AGGNN	98.18 ± 0.10	94.72 ± 0.29	-	70.26 ± 0.38	80.03 ± 0.22	-
Text GCN	0.9707 ± 0.0010	0.9356 ± 0.0018	0.8634 ± 0.0009	0.6836 ± 0.0056	0.7674 ± 0.0020	-
TensorGCN	0.9804 ± 0.0008	0.9505 ± 0.0011	0.8774 ± 0.0005	0.7011 ± 0.0024	0.7791 ± 0.0007	-
RGNN	98.82	95.28	87.78	-	83.87	-
TextGNN	97.8 ± 0.2	94.6 ± 0.3	-	69.4 ± 0.6		-
ReGNN	97.93 ± 0.31	95.17 ± 0.17	-	67.93 ± 0.33	78.71 ± 0.56	-
HINT	98.12±0.09	95.02± 0.18	-	68.79± 0.12	77.03±0.12	-

Method	R8	R52	20NG	Ohsumed	MR	IMDb
ST-Text-GCN	-	-	-	0.4242	0.7244	-
ME-GCN	0.8679	0.7828	0.2861	0.2740	0.6811	-
DADGNN	98.15±0.16	95.16±0.22	-	-	-	88.49±0.59
LSTM-GAT	97.85 ± 0.32	94.74 ± 0.41	-	70.23 ± 0.41	78.04 ± 0.53	90.32 ± 0.22
Text MGNN	0.9739	0.9420	-	0.7000	0.7466	-

For general text classification tasks, Accuracy, Precision, Recall and varying F1 are commonly used evaluation metrics for comparing with other baselines. However, for GNN based models, only representing the model performance cannot effectively represent the multi-aspects of proposed models. In this case, there are many papers conducting external processes to evaluate and analyse the GNN based classifier from multiple views.

Chapter 5

5.0 Conclusion and Future Scope

5.1 Conclusion

This survey explores the application of Graph Neural Network in text classification and provides a detailed discussion of how these GNN models are designed and highlights the datasets commonly used for their training and evaluation. Corpus level GNN focus on modeling relationships across the entire corpus of documents, building a global graph structure where both words and documents are formed based on various forms of co-occurrence of similarity. Document level GNN focuses on building a graph structure within individual documents. This method gives a deeper understanding of the document's internal structure. GNN are able to model the interaction between words and documents more effectively than other models. This is beneficial in capturing global context at the corpus level and local context within individual documents. It provides a flexible structure that can represent relationships beyond simple word co-occurrences, incorporating various such as syntactic relationships, word dependencies and other semantic information.

5.2 Challenges and Future Work

Graph Neural Network has been increasingly applied to text classification tasks, but several challenges remain in improving their performance, scalability and applicability to real world scenarios. This summary highlights the main areas of the future work.

Text classification methods have seen significant improvements with the development of pre-trained models like BERT, GPT and prompt-learning methods. These models set a high level for performance. GNN based models for text classification, which do not benefit from such pre-training, struggle to achieve similar levels of accuracy. Future research should focus on way to combine GNN with pre-trained models to leverage the benefits of both. This hybrid approach could potentially improve text classification by combining GNN's ability to model relational structures with the strong contextual understanding provided by pre-trained language models.

Existing GNN models for text classification are relatively simple in terms of the types of graphs, they use. More advanced Graph structures such as heterogeneous graphs that capture richer relationships between words and documents could enhance model performance. Exploring different types of

graphs and their applications to text classification is a key area for future work.

While corpus-level GNNs for text classification have demonstrated strong performance, they typically operate in a transductive setting, where the entire graph is available during training. This limits their applicability to real-world tasks, where models must handle unseen data. In inductive learning methods, where models generalize to new unseen data, need to be further exploration. Constructing large-scale graphs requires significant computational resources and current methods are not well optimized for these models. Improving the scalability of GNNs for training and testing, especially in inductive settings is a critical area for future development.

References

1. Ikonomakis, M., Sotiris Kotsiantis, and Vasilis Tampakas. "Text classification using machine learning techniques." *WSEAS transactions on computers* 4, no. 8, pp. 966-974, 2005.
2. Korde, Vandana, and C. Namrata Mahender. "Text classification and classifiers: A survey." *International Journal of Artificial Intelligence & Applications* 3, no. 2, 85, 2012.
3. Taha, Kamal, Paul D. Yoo, Chan Yeun, Dirar Homouz, and Aya Taha. "A comprehensive survey of text classification techniques and their research applications: Observational and experimental insights." *Computer Science Review* 54, 100664, 2024.
4. Kowsari, Kamran, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. "Text classification algorithms: A survey." *Information* 10, no. 4, 150, 2019.
5. Dasgupta, Anirban, Petros Drineas, Boulos Harb, Vanja Josifovski, and Michael W. Mahoney. "Feature selection methods for text classification." In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 230-239, 2007.
6. Dumais, Susan, John Platt, David Heckerman, and Mehran Sahami. "Inductive learning algorithms and representations for text categorization." In *Proceedings of the seventh international conference on Information and knowledge management*, pp. 148-155, 1998.
7. Li, Chen, Xutan Peng, Hao Peng, Jianxin Li, and Lihong Wang. "TextGTL: Graph-based Transductive Learning for Semi-supervised Text Classification via Structure-Sensitive Interpolation." In *IJCAI*, pp. 2680-2686, 2021.
8. Khan, Aurangzeb, Baharum Baharudin, Lam Hong Lee, and Khairullah Khan. "A review of machine learning algorithms for text-documents classification." *Journal of advances in information technology* 1, no. 1, pp. 4-20, 2010.
9. Zijian Liang, Hui Ding, and Wenlong Fu. A Survey on Graph Neural Networks for Recommendation. In *2021 International Conference on Culture-oriented Science & Technology (ICCST)*. IEEE, pp. 383–386, 2021.
10. Maas, Andrew, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. "Learning word vectors for sentiment analysis." In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142-150, 2011.

11. Wu, Zonghan, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S. Yu Philip. "A comprehensive survey on graph neural networks." *IEEE transactions on neural networks and learning systems* 32, no. 1, pp. 4-24, 2020.
12. Mirończuk, Marcin Michał, and Jarosław Protasiewicz. "A recent overview of the state-of-the-art elements of text classification." *Expert Systems with Applications* 106, pp. 36-54, 2018.
13. Benamira, Adrien, Benjamin Devillers, Etienne Lesot, Ayush K. Ray, Manal Saadi, and Fragkiskos D. Malliaros. "Semi-supervised learning and graph neural networks for fake news detection." In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 568-569, 2019.
14. Cavnar, William B., and John M. Trenkle. "N-gram-based text categorization." In *Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval*, vol. 161175, pp. 14, 1994.
15. Deng, Xuelian, Yuqing Li, Jian Weng, and Jilian Zhang. "Feature selection for text classification: A review." *Multimedia Tools and Applications* 78, no. 3, pp. 3797-3816, 2019.
16. Li, Nian, Yunzhu Pan, Chen Gao, Depeng Jin, and Qingmin Liao. "Full-stage Diversified Recommendation: Large-scale Online Experiments in Short-video Platform." In *Proceedings of the ACM on Web Conference 2024*, pp. 4565-4574, 2024.
17. Hakim, Ari Aulia, Alva Erwin, Kho I. Eng, Maulahikmah Galinium, and Wahyu Muliady. "Automated document classification for news article in Bahasa Indonesia based on term frequency inverse document frequency (TF-IDF) approach." In *2014 6th international conference on information technology and electrical engineering (ICITEE)*, pp. 1-4. IEEE, 2014.
18. Kadhim, Ammar Ismael. "Survey on supervised machine learning techniques for automatic text classification." *Artificial intelligence review* 52, no. 1, pp. 273-292, 2019.
19. Khan, Aurangzeb, Baharum Baharudin, Lam Hong Lee, and Khairullah Khan. "A review of machine learning algorithms for text-documents classification." *Journal of advances in information technology* 1, no. 1, pp. 4-20, 2010.
20. Shah, Foram P., and Vibha Patel. "A review on feature selection and feature extraction for text classification." In *2016 international conference on wireless communications, signal processing and networking (WiSPNET)*, pp. 2264-2268. IEEE, (2016).
21. Vijayan, Vikas K., K. R. Bindu, and Latha Parameswaran. "A comprehensive study of text classification algorithms." In *2017*

International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1109-1113. IEEE, 2017.
22. Wang, Jingyu. "Text classification based on GNN." In *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, pp. 94-97. IEEE, 2020.
23. Zhang, Yufeng, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. "Every document owns its structure: Inductive text classification via graph neural networks." *arXiv preprint arXiv:2004.13826*, 2020.
24. Deng, Zhaoyang, Chenxiang Sun, Guoqiang Zhong, and Yuxu Mao. "Text classification with attention gated graph neural network." *Cognitive Computation* 14, no. 4, pp. 1464-1473, 2022.
25. Yao, Liang, Chengsheng Mao, and Yuan Luo. "Graph convolutional networks for text classification." In *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, pp. 7370-7377, 2019.
26. Liu, Xien, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. "Tensor graph convolutional networks for text classification." In *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 05, pp. 8409-8416, 2020.
27. Liu, Chongyi, Xiangyu Wang, and Honglei Xu. "Text Classification Using Document-Relational Graph Convolutional Networks." *IEEE Access* 10, pp. 123205-123211, 2022.
28. Sun, Zhongtian, Anoushka Harit, Alexandra I. Cristea, Jialin Yu, Lei Shi, and Noura Al Moubayed. "Contrastive learning with heterogeneous graph attention networks on short text classification." In *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1-6. IEEE, 2022.
29. Wei, Xinde, Hai Huang, Longxuan Ma, Ze Yang, and Liutong Xu. "Recurrent graph neural networks for text classification." In *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 91-97. IEEE, 2020.
30. Huang, Lianzhe, Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. "Text level graph neural network for text classification." *arXiv preprint arXiv:1910.02356,* 2019.
31. Meng, Yining, Guoyin Wang, and Qun Liu. "Multi-layer convolutional neural network model based on prior knowledge of knowledge graph for text classification." In *2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, pp. 618-624. IEEE, 2019.
32. Zong, Daoming, and Shiliang Sun. "GNN-XML: graph neural networks for extreme multi-label text classification." *arXiv preprint arXiv:2012.05860,* 2020.

33. Li, Wei, Shuheng Li, Shuming Ma, Yancheng He, Deli Chen, and Xu Sun. "Recursive graphical neural networks for text classification." *arXiv preprint arXiv:1909.08166,* 2019.
34. Zhang, Chong, He Zhu, Xingyu Peng, Junran Wu, and Ke Xu. "Hierarchical information matters: Text classification via tree based graph neural network." *arXiv preprint arXiv:2110.02047,* 2021.
35. Cui, Hongyan, Gangkun Wang, Yuanxin Li, and Roy E. Welsch. "Self-training method based on GCN for semi-supervised short text classification." *Information Sciences* 611, pp.18-29, 2022.
36. Wang, Kunze, Soyeon Caren Han, Siqu Long, and Josiah Poon. "Me-gcn: Multi-dimensional edge-embedded graph convolutional networks for semi-supervised text classification." *arXiv preprint arXiv:2204.04618,* 2022.
37. Liu, Yonghao, Renchu Guan, Fausto Giunchiglia, Yanchun Liang, and Xiaoyue Feng. "Deep attention diffusion graph neural networks for text classification." In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 8142-8152, 2021.
38. Wang, Ziyun, Xuan Liu, Peiji Yang, Shixing Liu, and Zhisheng Wang. "Cross-lingual text classification with heterogeneous graph neural network." *arXiv preprint arXiv:2105.11246,* 2021.
39. Wu, Man, Shirui Pan, Xingquan Zhu, Chuan Zhou, and Lei Pan. "Domain-adversarial graph neural networks for text classification." In *2019 IEEE international conference on data mining (ICDM)*, pp. 648-657. IEEE, 2019.
40. Gu, Yongchun, Yi Wang, Heng-Ru Zhang, Jiao Wu, and Xingquan Gu. "Enhancing text classification by graph neural networks with multi-granular topic-aware graph." *IEEE Access* 11, pp. 20169-20183, 2023.
41. Wang, Haitao, and Fangbing Li. "A text classification method based on LSTM and graph attention network." *Connection Science* 34, no. 1, pp. 2466-2480, 2022.
42. Gao, Tianyu, Adam Fisch, and Danqi Chen. "Making pre-trained language models better few-shot learners." *arXiv preprint arXiv:2012.15723,* 2020.
43. Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532-1543, 2014.
44. Zhang, Wen, Taketoshi Yoshida, and Xijin Tang. "A comparative study of TF* IDF, LSI and multi-words for text classification." *Expert systems with applications* 38, no. 3, pp. 2758-2765, 2011.
45. Mikolov, Tomas. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* 3781, 2013.

46. Joachims, Thorsten. "Text categorization with support vector machines: Learning with many relevant features." In *European conference on machine learning*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 137-142, 1998.
47. Lang, Ken. "Newsweeder: Learning to filter netnews." In *Machine learning proceedings 1995*, Morgan Kaufmann, pp. 331-339, 1995.
48. Li, Chen, Xutan Peng, Hao Peng, Jianxin Li, and Lihong Wang. "TextGTL: Graph-based Transductive Learning for Semi-supervised Text Classification via Structure-Sensitive Interpolation." In *IJCAI*, pp. 2680-2686. 2021.
49. Liu, Pengfei, Xipeng Qiu, and Xuanjing Huang. "Recurrent neural network for text classification with multi-task learning." *arXiv preprint arXiv:1605.05101,* 2016.
50. Pang, Bo, and Lillian Lee. "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales." *arXiv preprint cs/0506075,* 2005.
51. Li, Qian, Hao Peng, Jianxin Li, Congying Xia, Renyu Yang, Lichao Sun, Philip S. Yu, and Lifang He. "A survey on text classification: From shallow to deep learning." *arXiv preprint arXiv:2008.00364,* 2020.