Leveraging Graph Neural Networks for Text Classification with Semantic and Structural Insights

Remya R. K. Menon[a], Jyothish S L[b] and Ajith B. T. K.[c]
Department of Computer Science and Applications, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Amritapuri, India

Keywords: Graph Neural Networks, Text Classification, Graph Auto Encoders, Node Embeddings, Sentence-BERT, Natural Language Processing, Contextual Similarity.

Abstract: Applications which involve text classification may still need a breakthrough in capturing the latent structure in the text and more complex dependencies which limits its capacity to make correct predictions. This paper presents a new approach to a text classification application in which a hybrid graph representation learning algorithm has been used to demonstrate interactions between latent semantic and structural data in text documents. Text is represented as a graph, where a node represents a sentence and an edge represents the semantic relationship between two nodes. With nodes converted to embeddings generated through Sentence-BERT, it offers contextualized representations for every node. Along with this framework, we also learn low-dimensional representations of the text graphs using graph auto-encoders. Our model thus enhances generalization and has a powerful representation for downstream tasks by minimizing the difference between reconstructed and input graphs. Experimental results demonstrate that our model surpasses traditional methods by successfully integrating semantic and structural information to enhance classification accuracy. This work contributes to the advancement of GNN-based architectures for text retrieval, demonstrating the potential of graphs in natural language processing.

1 INTRODUCTION

Text classification is a very well-known process that can be directly used major process like sentiment analysis, spam detection, document categorization and constructing a recommendation system. There are various algorithms for this process. In order to improve the accuracy of these algorithms, text preprocessing plays a major role. Capturing the latent sense of the text has been a well-known research problem. Text representations also help to improve the evaluation metrics. From the tf-idf model to deep learning language models, representations have been the primary focus in improving the classification accuracy. Our work focuses on a retrieval application which requires a strong representation and a classifier that predicts the relevance of a document for a given query. For this purpose, we are utilizing the strength of graph neural networks (GNNs), which are representation learning models used to represent the documents as nodes of a graph. This permits the model to comprehend complicated graph relationships by utilizing message passing to collect information from neighboring nodes. GNNs excel in text classification, link prediction, and clustering tasks. Graph Neural Networks (GNNs) represent a modern shift from machine learning and artificial intelligence to deep learning concepts. With the increase in related data arising from numerous domains including social media, recommendation systems, biology, and cyber security, GNNs have significantly improved. These models are able to extract complex relationships, which makes them pivotal in the design of real-world problems. Consequently, Graph Neural Networks (GNNs) has the ability to apprehend the complicated structural nuances hidden inside the text. Those insights, comprising dependencies, connections, and contextual relationships, are capable of feature extraction and predictive tasks. Consequently, GNNs have proliferated across diverse applications, revolutionizing the machine learning capabilities.

[a] https://orcid.org/0000-0001-7365-9058
[b] https://orcid.org/0009-0008-6262-6208
[c] https://orcid.org/0009-0003-8116-6810

R. K. Menon, R., S L, J. and B. T. K., A.
Leveraging Graph Neural Networks for Text Classification with Semantic and Structural Insights.
DOI: 10.5220/0013587700004664
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 3rd International Conference on Futuristic Technology (INCOFT 2025) - Volume 2*, pages 93-104
ISBN: 978-989-758-763-4
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

2 RELATED WORK

GNN has witnessed rapid development in addressing the unique demanding situations where data are presented as graphs at places where traditional deep learning approaches often fail to provide significant insight. This comprehensive survey on GNNs offers an in-depth analysis that includes critical aspects along with the basics of GNN, the interaction with convolution neural networks, GNN message-passing mechanisms, various GNN models and suitable applications. Inside the message-passing mechanism of a neural network, every node has its message stored in the form of characteristic vectors(Khemani et al., 2024). This process aggregates the vectors to create a new message. Graphs can be classified as directed or undirected, static or dynamic, homogeneous or heterogeneous, transductive or inductive. In (Yuan et al., 2023), GCN and GAT / GAN are compared with respect to the processes incolved. GCN entails initialization, convolution operation, weighted aggregation, activation feature, and stacking. GAT/GAN consists of initialization, self-attention mechanism, attention computation, weighted aggregation, more than one attention, output mixture, learning weights, and stacking layers. These models have applications in graph construction, social networks, and citation networks.

Document preprocessing is always an important step in document classification. (Kavitha et al., 2023) has used mutual information for feature extraction based on word sense disambiguation. This method claims to improve the text classification by distinguishing the sense of polysemy words correctly. Sparse Graph Auto-Encoders have shown remarkable contribution to improving the performance of document recommendation systems as proved by (Menon et al., 2023). Explainability is one of the two vital topics of interest these days.In the paper, (Li et al., 2022), a comprehensive assessment of contemporary GNN explainability strategies is presented, including evaluations of quantitative metrics and datasets. Furthermore, the paper introduces a novel evaluation metric for comparing various GNN explainability techniques using unique real-world datasets, GNN architectures, and future instructions for GNN explainability. In explainability, the two primary modern methods are function visualization and behavior approximation(Li et al., 2022). Function visualization encompasses techniques such as saliency maps for images and heatmaps for text, which highlight key regions or words contributing to predictions. However, these methods encounter challenges when applied to non-Euclidean data structures, such as graphs, and can involve subjective evaluation. Behavior approximation, on the other hand, relies on interpretable models designed to replicate the behavior of black-box systems.The evaluation of modern explanation methods revolves around two main criteria: plausibility and correctness. Plausibility refers to how convincing the explanations are to humans, often relying on subjective human judgment. Correctness, on the other hand, assesses whether an explanation accurately reflects the reasoning process of the underlying model, with various metrics proposed for this evaluation(Li et al., 2022).

Explainability methods are generally divided into two categories: those that originate outside of GNNs and those specifically developed for GNNs. GNN-specific strategies often adapt gradient-based and decomposition-based methods to explain graph neural networks. Examples of such techniques include GNN Explainer and PGExplainer(Parameterized Explainer), which aim to generate explanations by identifying important sub-graphs, DeepLIFT, GNN-LRP, Grad-CAM, SubgraphX, and XGNN.These are provided in pytorch geometric. Other methods, like Graph Mask and SubgraphX, provide both instance-level and global explanations by effectively discarding unnecessary edges or exploring diverse sub-graphs. XGNN offers a model level clarification with the aid of producing graph patterns for class predictions(Yuan et al., 2023).

Explainable AI tools are used in (Reghu et al., 2024) to interpret the output produced by retrieval systems. It has a classifier as a sub-task which predicts the relevance of a document for a given query. The results of the evaluation metrics for this system are explained using various tools like LIME, SHAP, Partial Dependency Plots, DALEX, Anchors and saliency maps.

The critical significance of evaluating the quality and reliability of factors generated by using graph neural networks (GNNs) in diverse high-stake applications is given in (Agarwal et al., 2022). It emphasizes the need for standardized evaluation techniques and reliable information sources to assess GNN correctly. The authors introduce ShapeGN (Shape Generation Networks), an artificial graph data generator, and GraphXAI, a graph explainability library, as gear to aid the benchmarking modern GNN explainers. These resources enhance explainability research in GNNs with the aid of providing a broader surroundings for evaluating post-hoc motives throughout numerous real-world packages.

A modern approach to text summarization with the usage of Graph Neural Networks (GNNs) and Named Entity Recognition (NER) models is presented in (Khan et al., 2024). The paper highlights the challenges in comparing text summarization systems because of the subjective nature of defining a summary measure and the limitations of widely used metrics. It emphasizes the importance of capturing the context in summarization and the need for resource-efficient summarization. The paper additionally mentions the significance of key entities in text for effective summarization and the enhancements in extractive text summarization methodologies. Furthermore, it describes the process of sentence selection and redundancy elimination to produce concise and informative summaries. The evaluation and testing phase includes modern metrics like ROUGE and user studies to assess the quality and applicability of the generated summaries. The combination of NER and GNNs enhances the performance and relevance of text summarization methods by handling the large amount of textual data available today.

(Zhang et al., 2020) proposes a novel method called TextING for inductive text classification with the help of Graph Neural Networks (GNNs). Conventional text classification methods fails in capturing contextual word relationships for new words. TextING addresses those troubles by means of constructing personalized graphs and cutting-edge fine-grained phrase representations based on their local contexts. This technique allows the model to generate embeddings for unseen words efficiently. Large experiments on four benchmark datasets exhibit that TextING outperforms traditional text based techniques. The paper highlights three key contributions: proposing a new GNN model for text classification that captures text-level word interactions, generalizing the model to deal with new, unseen words throughout the testing, and demonstrating the superior performance of the model via substantial experiments.

The rise of social networking models and the resulting abundance of time-sensitive news facts, which has significant economic value for companies undertaking data mining and sentiment analysis is discussed in (Li et al., 2024). Text classification, a key research area, has evolved from traditional machine learning techniques like Naive Bayesian (NB), Support Vector Machine (SVM), and Maximum Entropy models, to deep learning methods that automatically extract features and capture semantic statistics. The paper highlights the constraints of existing models, together with TextCNN's lack of ability to represent local textual data and contextual relationships. It additionally reviews improvements in models incorporating the attention mechanism and Graph Neural Networks (GNNs), like Graph Attention Networks (GAT). The study proposes a unique text classification model using GATs that integrates lexical knowledge, carries noise perturbations for adverse training to improve robustness, and employs a multi-head attention mechanism to enhance classification accuracy.

(Rastakhiz et al., 2024) introduces a method for text classification using Graph Neural Networks (GNNs). The proposed method includes converting raw text into structured heterogeneous graphs, which effectively capture complex data relationships. By transforming each document into a graph and capturing both explicit and implicit contextual information, the text classification problem is framed as a graph classification problem. This method is adaptable to texts of any length, eliminating the need to set the maximum lengths or padding shorter texts.

The study evaluates the models using two datasets: Yelp Polarity for binary sentiment analysis and AG News for multiclass text classification. The outcomes spotlight the effectiveness of the use of dependencies and tags to enhance the model's contextual understanding. Compared to conventional baselines, the GNN-based technique demonstrates advanced text representation abilities, underscoring the potential of GNNs to improve text classification accuracy and robustness.

The importance of text classification in Natural Language Processing (NLP), including topic classification and sentiment analysis is outlined in (Wang et al., 2023). Traditional text classification methods make use of N-gram or Term Frequency-Inverse Document Frequency (TF-IDF) representations combined with Machine Learning models like SVM. With the advent of neural networks, more advanced models like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and attention-based models have been employed. However, these models often fail to capture complicated word-document relationships and effectively identify contextual word relationships. To address these challenges, Graph Neural Networks (GNNs) have been introduced, utilizing the graph-structured data to enhance text classification.

(Zhou et al., 2018) highlights applications of GNN in social networks, in which they enhance tasks like community detection and link prediction; knowledge graphs, where they improve reasoning over entities and relationships for applications like query answering; biological networks, in which they analyze molecular structures and protein interactions for drug discovery and genomics; and recommender systems, where they capture the patterns in user-item interaction to improve recommendation accuracy.Key methodologies discussed consist of GCNs, which expand convolution operations to graph data for effective information propagation; GATs, which use attention mechanisms to focus on relevant graph elements during information aggregation; and GRNs, which uses recurrent neural networks to capture temporal dependencies in dynamic graphs. Moreover, the paper identifies possibilities for future research, along with developing scalable GNN models, improving model interpretability, advancing methods for dynamic graphs, and integrating GNNs with other data modalities to create more robust models. The paper underscores the potential of GNNs across numerous domains, encouraging continued research to address the challenges and leverage rising possibilities.

(Menon et al., 2020) has built a complete retrieval system using two models viz. Kernel pooling based neural ranking model and semantic similarity based model. There are three layers in Kernel pooling model including a representation layer where documents are embedded using neural models, a kernel pooling layer and a ranking layer. Semantic similarity based model uses Word movers distance and cosine similarity as methods to find the similarity between documents and queries.Cranfield, Medline and WikiQA collection has been used for evaluation.

Text Graph Convolutional Neural Networks (TextGCN) and Vision Transformers (ViT) have been used in (Visweswaran et al., 2024) for fake news identification in online posts which contain both text and images. TextGCN has outperformed SVM and Random forest in precision and recall. The effectiveness of graph model in classifying Telugu news content into different topics is done in (Namburu et al., 2024). They found that bi-directional LSTM performed better in their experimental environment where limited power of BERT were utilized for representation and Parts of Speech relationships were not included. In conclusion, our paper throws light into the broad aspects of existing literature related to graphs in the context of its different architectures, pre-processing, explainability, evaluation, applications and domains.

3 PROPOSED SYSTEM

The proposed approach leverages Graph Neural Networks (GNNs) and Graph Autoencoders (GAEs) to improve text classification by considering the latent semantic and structural information in the text. This methodology is designed to improve classification accuracy by integrating these two aspects effectively.

3.1 Architecture

The architecture depicted in the figure 1) represents a pipeline designed for training and testing a Graph Neural Network (GNN) model, specifically a GAE (Graph Autoencoder)-based framework. The input data comprises three key components: Query, Abstract, and Relevance, which are utilized throughout the process to model and predict relevance relationships.

3.1.1 Data Description and Preprocessing

The dataset used for our work includes abstracts, queries and the relevance information of abstracts for different queries.In the preprocessing step, the dataset is loaded, and key components—queries, abstracts, and relevance labels—are extracted .Then stopword removal is also done.

3.1.2 Latent Semantics through Node Embeddings

The queries and abstracts are encoded into dense vector embeddings that capture their semantic meaning. Those embeddings are generated by using pretrained language models, such as BERT or Sentence-Transformers, encapsulating the contextual statistics for each phrase. This enables the model to accommodate complexities in natural language, including polysemy (meaning of one word as more than one word) and synonymy (words with similar meanings). These embeddings then formulate the semantic richness in the model.

3.1.3 Text Representation as Graphs

The embeddings are used to create the adjacency matrix .This matrix is then used to construct the graph. In this case, the edges between nodes denote relationships that may be extracted from techniques such as co-prevalence or dependency parsing. Thus, this graph-based model captures nearby and global structural elements which enable representations to be used in natural language processing tasks.

3.1.4 Graph Auto Encoder (GAE) Architecture

A Neural Network architecture designed to learn a low-dimensional representation of a graph in an unsupervised manner.

Goal: To encode the input into a lower-dimensional representation, then reconstruct the original graph from this representation.

GAE consists of two main components:

1.Encoder

Input: A graph represented as an adjacency matrix.The graph passes through multiple Graph Convolutional Network (GCN) layers, with each layer pro-

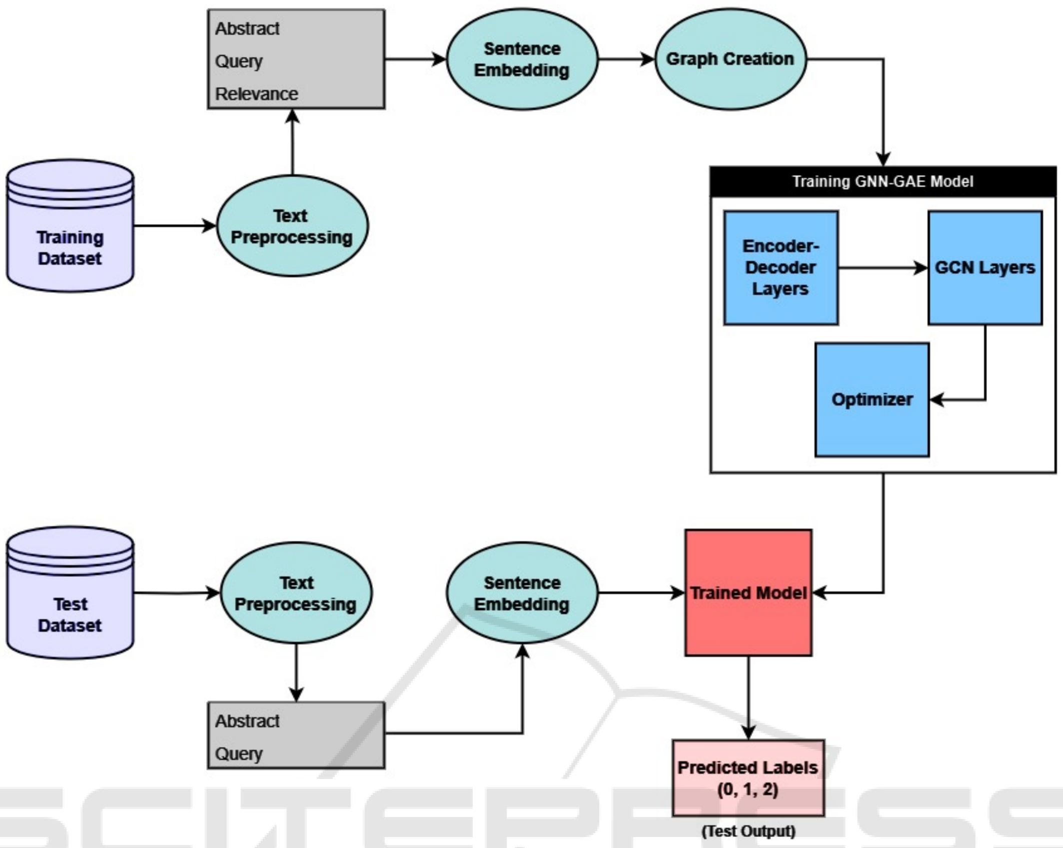


Figure 1: Architecture of the System.

ducing a new representation of the graph by performing nonlinear or convolutional operations on the input graph. This allows the model to capture the relevant features of the graph and transform them into a lower-dimensional space.

Output: A low-dimensional matrix known as the latent space representation that captures the essential features of the input graph to reconstruct the original graph.

2.Decoder

Input: The low-dimensional representation (embeddings) produced by the encoder. The encoded representation is passed through a series of fully connected layers, where each layer applies a non-linear transformation to the input and outputs a new set of features.

Output: The reconstructed graph, which is as close as possible to the input graph. The reconstruction is evaluated using a loss function that measures the difference between the reconstructed graph and the original input graph, such as the mean square error (MSE).

3.1.5 Graph Neural Networks for Text Classification

The final node representations are used for category classification. A classifier, possibly a fully connected layer, classifies the labels based on the representations this model learns. Using the semantic as well as the structural features, the model is accurate and has better performance compared to traditional text classification approaches using just sequential steps alone.

3.2 Algorithm

The proposed algorithm introduces an innovative approach to text classification that uses a Graph Auto-Encoder (GAE) model. Our work constructs a graph where nodes represent abstracts and queries. Edges signify their semantic relationships. The algorithm processes this graph to learn a low-dimensional latent representation which captures the essential features of the text. In addition to learning this latent representation, the GAE reconstructs the original graph, ensuring that the model retains key information about the connections between abstracts and queries. This reconstruction step helps to refine the representation, enhancing its ability to capture meaningful patterns in the data.

Input: Corpus of abstracts and Queries

Output: Relevance of abstracts for different queries

Step 1: Preprocessing Preprocessing is a critical step in preparing the text for semantic embedding. This level includes cleaning and normalizing the text to enhance the generated embeddings.Preprocess the entire document set, including tokenization and removing stop words. stemming/lemmatization has not been attempted as it may lose the phrasal presence in the text.In the merged dataset, a few documents may not have abstracts, or a few queries may additionally lack narratives. These lacking values are treated by filling in placeholder textual content for missing narrative facts and removing rows with lacking abstracts. This ensures that the dataset is suitable for similarity check.

Step 2: Adjacency Matrix Construction

- Generate both abstract and query embeddings.
- Values of the matrix stores the similarity scores between abstract-abstract and abstract-query.

Step 3: Graph Construction Construct a graph from the adjacency matrix where:

- *Nodes* represent both abstract and query embeddings.
- *Edges* represent the connections between them based on abstract-abstract and abstract-query relationships.
- *Weights* on the edges are determined by:
  - similarity scores

Step 4: GAE Model Pass the graph object as input to the Graph Autoencoder (GAE) model.

Encoder Network The encoder network maps the input graph to a low-dimensional latent representation. The encoder consists of:

- Multiple layers of graph convolution layers.
- Fully connected layers.
- Relu Activation functions.

Low-Dimensional Representation The output of the encoder is a low-dimensional representation of the input graph, capturing the structural properties of the graph.

Decoder Network The decoder receives the latent representation and generates a reconstruction of the original graph.

Step 5: Loss Calculation and weight updates A loss function measures the difference between the reconstructed graph and the original graph. The objective is to minimize this difference to learn a good latent representation. During backpropagation, the weights and biases of both the encoder and decoder are updated to minimize the loss function.

Step 6: GNN-based Classification A dataset comprising of query embeddings, abstract embeddings and corresponding label is constructed. Here labels are 0 (Non-relevant),1 (partially relevant) and 2 (relevant). This data is given for training to a GNN classifier which then predicts the relevance of an abstract for a query.

4 EXPERIMENTAL RESULTS AND ANALYSIS

4.1 CORD-19 Dataset

Cord-19 which is a biomedical dataset is used in this work. It contains full-text articles, abstracts, and metadata associated with COVID-19. This dataset serves as a foundation for the evaluation of classification models.

4.1.1 Key Files and Their Roles

- **topics-rnd3**: A CSV record containing the topics, each related to a completely unique subject matter, query, and narrative. These queries are used for searching relevant documents.
- **docids-rnd3**:This report includes a list of document IDs which can be potential candidates for relevance matching.
- **qrels**: A CSV file containing relevance judgments that suggest the level of relevance of each document to precise queries, based on previous evaluations.

This work deals with the identification of abstracts that are applicable to queries from the cord-19 dataset. The evaluation of relevance is entirely based on the cosine similarity between the query and abstract embeddings.

4.2 Data Loading and Merging

The raw information is loaded. The records are then merged such that every query is associated with its corresponding abstracts, ensuring that every question has a related set of abstracts to evaluate for relevance.

4.3 Embedding Generation and Graph Construction

The Sentence-BERT model, specifically the `paraphrase-MiniLM-L6-v2` variant, is used to generate semantic embeddings for both the queries and the abstracts. Sentence-BERT is a state-of-the-art model designed for generating high-quality embeddings that capture the semantic meaning of sentences, making it ideal for document retrieval tasks.The core idea of the proposed model is to treat the query-abstract pairs as nodes in a graph Figure 2), where each node represents either a query or an abstract. The relationships between these nodes are learned using a GNN, which processes the graph structure and node features (embeddings of the queries and abstracts).

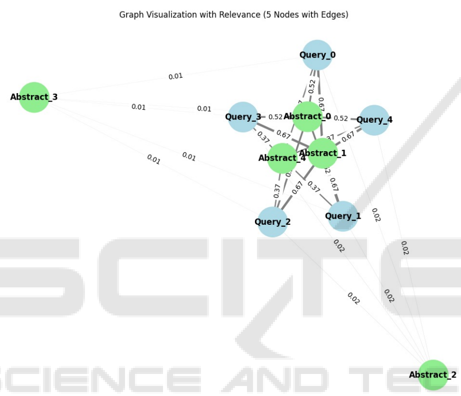


Figure 2: Query-abstract relationship graph.

4.4 Training Using Graph Neural Network (GNN)

The model is trained using a GAE and then a Graph Neural Network (GNN) to categorise files based on their semantic capabilities. The graph structure was constructed using cosine similarity between abstracts, and a threshold of 0.5 applied to edges among abstracts with moderate similarity. For training, the GNN utilizes graph convolutional layers (GCNConv), accompanied by a completely connected layer for binary class. The model is trained with a standard cross-entropy loss and an Adam optimizer. The training technique involved minimizing the loss characteristic by backpropagating the gradients and updating the model parameters over 100 epochs. Assessment is performed at normal intervals (every 10 epochs), using class accuracy as the assessment metric. The model's last performance is assessed based on accuracy and the model's capability to classify unseen documents need to be considered. The final trained model is stored for future use.

Table 1: Classification Report of GNN

	Precision	Recall	F1-score	Support
Not Relevant	0.86	0.48	0.61	15191
Partially Relevant	0.21	0.48	0.29	2537
Relevant	0.26	0.57	0.36	2849
Accuracy			0.49	20577
Macro avg	0.44	0.51	0.42	20577
Weighted avg	0.70	0.49	0.54	20577

4.4.1 Results of GNN

The evaluation of the GNN's performance demonstrated its effectiveness in categorizing documents with high accuracy. Figure 1 shows the evaluation metrics across training epochs.

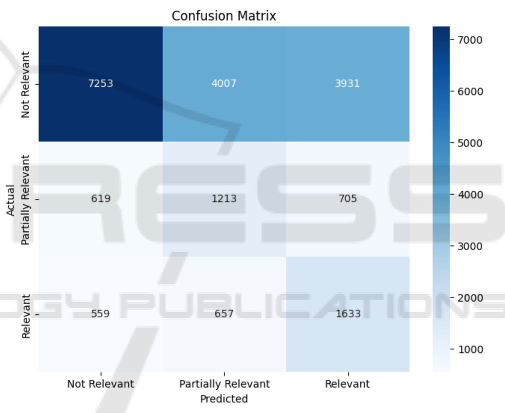


Figure 3: Confusion Matrix of GNN.

The confusion matrix (Figure 3) highlights the GNN model's classification performance across three categories: Not Relevant, Partially Relevant, and Relevant. The "Not Relevant" class shows the highest correct predictions (7251) but also a notable number of misclassifications into Partially Relevant (4007) and Relevant (3931), indicating that some irrelevant abstracts share overlapping features with more relevant ones. The "Partially Relevant" class proves the most challenging, with only 1213 correct predictions, while 619 samples were misclassified as Not Relevant and 705 as Relevant. This reflects the inherent ambiguity of the Partially Relevant class, as its abstracts often exhibit characteristics of both extremes. For the "Relevant" class, 1633 samples were correctly classified, but 559 were labeled as Not Relevant, and 657 as Partially Relevant, suggesting some overlap in semantic signals between relevance levels. Overall, the model performs well in distinguishing extremes (e.g., Not Relevant versus Relevant) but struggles with the intermediate Partially Relevant class due to overlapping features and subtle semantic boundaries. This analysis indicates that future improvements could focus on refining feature representations, addressing class imbalances, and enhancing the model's sensitivity to nuanced relevance levels.

4.4.2 Precision-recall graph of GNN

The Precision-Recall (PR) curve illustrates (Figure 4) the classification performance for three classes: "Not Relevant," "Partially Relevant," and "Relevant." The x-axis represents recall, which measures the model's ability to identify positive instances, while the y-axis shows precision, the proportion of correct positive predictions. The "Not Relevant" class (blue curve) demonstrates consistently high precision across different recall values. However, the "Partially Relevant" (orange curve) and "Relevant" (green curve) classes show a decline in precision as recall increases, indicating that the model has difficulty distinguishing these classes accurately. The curve highlights the varying performance across classes, which could indicate class imbalance or challenges in classification.

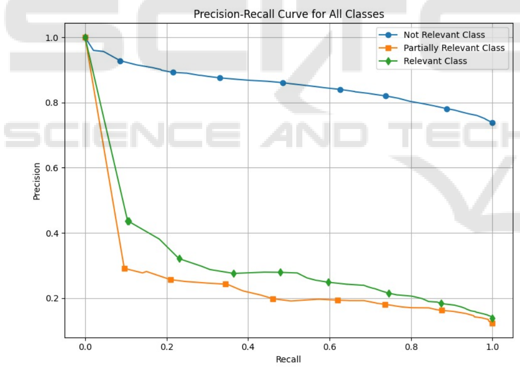


Figure 4: Precision-recall graph of GNN.

4.5 Training Using Graph Autoencoder (GAE)

A Graph Auto Encoder (GAE) is employed for low dimensional representation. Just like GNN training, the graph is built using cosine similarity between abstracts but the importance is shifted from classification to unsupervised representation learning. The GAE model consists of two graph convolutional layers, which analyze low-dimensional embeddings for each document based on the graph shape. These embeddings are used to reconstruct the adjacency matrix of the graph.

During training, the model optimizes the Mean Square Error (MSE) which may be credited to the reconstruction of the graph structure. The optimizer used is Adam, and the model is trained for one hundred epochs. The training losses and accuracies are recorded, and the final trained model is stored for future use.

Table 2: Classification Report of GAE

	Precision	Recall	F1-score	Support
Not Relevant	0.86	0.48	0.61	15191
Partially Relevant	0.21	0.43	0.28	2537
Relevant	0.25	0.62	0.36	2849
Accuracy			0.49	20577
Macro avg	0.44	0.51	0.42	20577
Weighted avg	0.70	0.49	0.54	20577

4.5.1 Results of GAE

The evaluation of the GAE's performance demonstrates its effectiveness in categorizing documents. (Table 2) shows the evaluation metrics across training epochs.

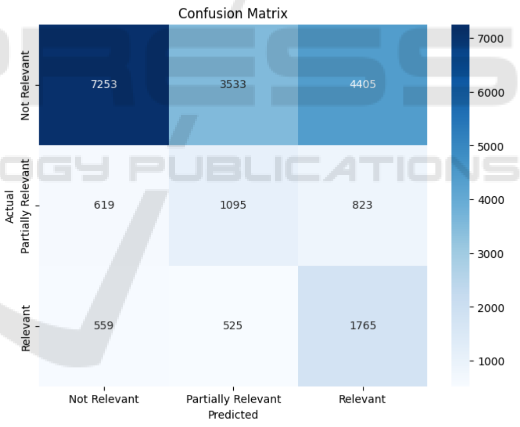


Figure 5: Confusion Matrix of GAE.

The confusion matrix (Figure 5)provides insight into the model's classification performance across three categories: Not Relevant, Partially Relevant, and Relevant. The Not Relevant class achieved the most correct predictions (7253), though a considerable number of samples were misclassified as Partially Relevant (3533) and Relevant (4405), suggesting some overlap in features with relevant abstracts. The Partially Relevant class proved more challenging, with 1095 correct predictions, while 619 were incorrectly labeled as Not Relevant and 823 as Relevant, highlighting the difficulty in differentiating this intermediate category. For the Relevant class, 1765 samples were classified correctly, but 559 were mistaken as Not Relevant and 525 as Partially Relevant, indicating some misalignment in recognizing subtle relevance cues. Overall, the model effectively distinguishes between the clear extremes (Not Relevant and Relevant) but struggles with the intermediate class due to overlapping features and subtle semantic similarities, pointing to areas for improvement in capturing nuanced distinctions.

4.5.2 Precision-recall graph of GAE

The graph (Figure 8) shows the Precision-Recall (PR) curves for three classes: ”Not Relevant,” ”Partially Relevant,” and ”Relevant.” The ”Not Relevant” class performs the best, maintaining high precision across all recall values. In contrast, the ”Relevant” and ”Partially Relevant” classes show a steep drop in precision and remain low as recall increases, indicating the model struggles to classify these two classes accurately. This suggests challenges such as class imbalance or overlapping features.

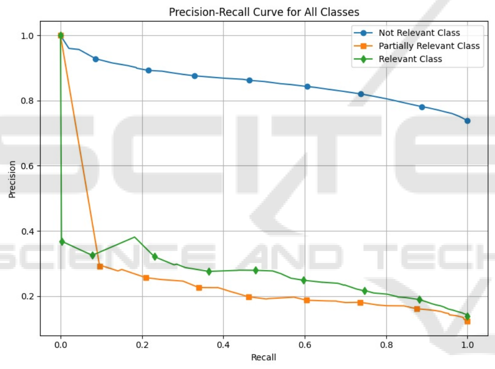


Figure 6: Precision-recall graph of GAE.

4.6 Training Using GNN With Encoder Decoder Layers

The core idea of the proposed model is to treat the query-abstract pairs as nodes in a graph, where each node represents either a query or an abstract. The relationships between these nodes are learned using a GNN, which processes the graph structure and node features (embeddings of the queries and abstracts).

The training process involves optimizing the model to predict the relevance score of a query-abstract pair. Relevance is classified into three categories: *Not Relevant*, *Partially Relevant*, and *Relevant*. These categories are represented as classes in the classification task.

The model utilizes a GNN to propagate information between nodes (queries and abstracts), allowing the system to learn not only from individual embeddings but also from the structural relationships between the queries and the abstracts. This interaction between the nodes is key to understanding the context of each query relative to the abstracts, and ultimately, predicting the relevance.

4.6.1 Layers of the Model

The proposed model consists of multiple layers that are designed to extract complex patterns from the graph:

- **Graph Convolutional Layers:** The model uses three layers of Graph Convolutional Networks (GCN). These layers apply graph convolution operations to the input embeddings, capturing the relationships between connected nodes. Each convolutional layer updates the node representations by aggregating features from neighboring nodes, thereby learning the dependencies between queries and abstracts.
- **Encoder Layer:** The output of the third GCN layer is passed through a fully connected encoder layer. This layer reduces the dimensionality of the features, preparing them for classification. The encoder layer is designed to capture the most relevant features from the graph-processed embeddings.
- **Decoder Layer:** The decoder layer takes the encoded features and maps them to the output space, which corresponds to the classes of relevance. The decoder layer applies a linear transformation to predict the relevance class of each query-abstract pair.
- **Dropout Layer:** A dropout layer is applied after each GCN layer to prevent overfitting and improve generalization by randomly setting a fraction of the input units to zero during training.

These layers work together to refine the node features iteratively and improve the accuracy of the final classification.

4.6.2 Prediction and Inference

After the model has been trained, predictions are made by passing the query-abstract embeddings through the graph neural network. The final output of the model is a predicted relevance score for each query-abstract pair, which corresponds to one of the three predefined classes: *Not Relevant*, *Partially Relevant*, or *Relevant*. The prediction process works as follows:

1. The query and abstract embeddings are processed by the GCN layers, where the node features are updated by aggregating information from neighboring nodes.
2. The encoded features are passed through the encoder layer, which extracts the most important features from the graph.
3. These features are then decoded by the decoder layer into a final class prediction.
4. The model's output is compared to the true label (ground truth), and the loss is computed during training. During inference, the class with the highest predicted score is chosen as the final label.

Table 3: Classification Report of GNN + Encoder Decoder

	Precision	Recall	F1-score	Support
Not Relevant	0.84	0.63	0.72	15191
Partially Relevant	0.24	0.35	0.29	2537
Relevant	0.27	0.53	0.36	2849
Accuracy			0.58	20577
Macro avg	0.45	0.50	0.45	20577
Weighted avg	0.69	0.58	0.61	20577

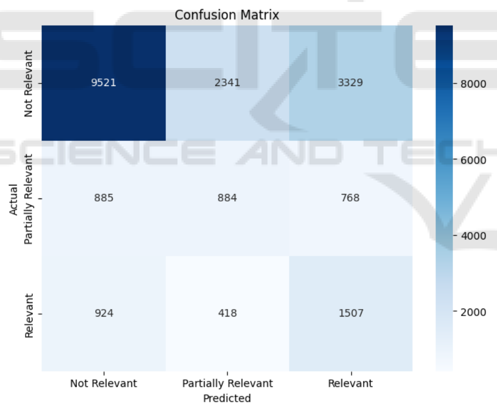


Figure 7: Confusion Matrix of GNN with Encoder-Decoder.

The confusion matrix (Figure7) provides insight into the model's performance across three categories: Not Relevant, Partially Relevant, and Relevant. The Not Relevant class achieved the highest accuracy, with 9,521 samples correctly classified, but still faced misclassification issues, with 2,341 predicted as Partially Relevant and 3329 as Relevant. For the Partially Relevant category, the model correctly identified only 884 samples, while 1,024 were misclassified as Not Relevant and 629 as Relevant, reflecting the challenge in distinguishing this intermediate class due to its overlapping characteristics with the others. The Relevant class had 1,509 correct predictions but saw 924 misclassified as Not Relevant and 418 as Partially Relevant, indicating some difficulty in identifying clear boundaries for relevance. Overall, while the model performs well in identifying Not Relevant samples, it struggles more with the intermediate Partially Relevant class and shows room for improvement in refining feature representation and better distinguishing between relevance levels.The predictions are evaluated using standard classification metrics, such as accuracy, precision, recall, and F1-score. These metrics help assess the model's performance in classifying the relevance of query-abstract pairs.

4.6.3 Precision-recall graph of GNN with Encoder Decoder

The graph illustrates the Precision-Recall (PR) curve for a model using a Graph Neural Network (GNN) combined with an encoder-decoder layer. It evaluates three classes: *Not Relevant* (blue), *Partially Relevant* (orange), and *Relevant* (green). The *Not Relevant* class demonstrates strong performance, maintaining precision above 70% across all recall values, which indicates the model's effectiveness in correctly identifying this category. However, the *Relevant* and *Partially Relevant* classes show a sharp drop in precision at low recall values and remain at lower levels as recall increases. This suggests that the model struggles to classify these ambiguous classes accurately, likely due to overlapping characteristics or class imbalance.

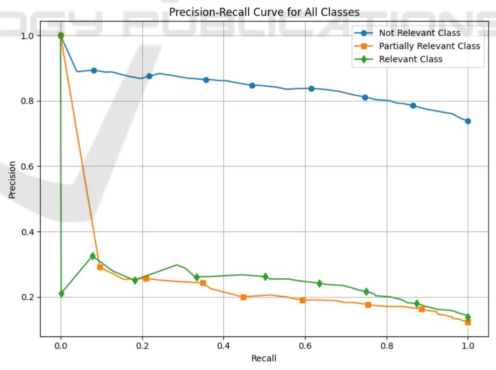


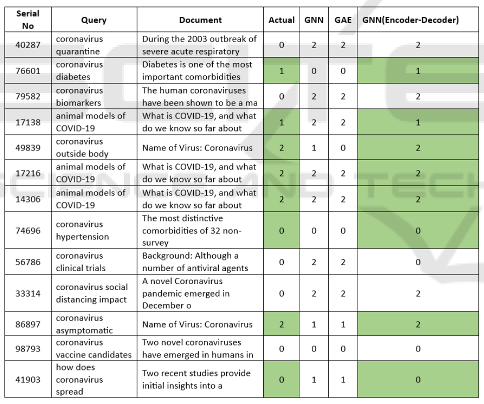
Figure 8: Precision-recall graph of GNN with Encoder Decoder.

5 MODEL PERFORMANCE COMPARISON: GNN, GAE, AND GNN WITH ENCODER-DECODER

In this section, we compare the performance of three models Graph Neural Network (GNN), Graph Autoencoder (GAE), and GNN with Encoder-Decoder in the task of classifying the relevance of query-document pairs. The task involves evaluating how well each model predicts the relevance of a document to a specific query. To conduct this evaluation, we used a subset of queries and their associated documents, considering the relevance of the documents as the actual labels. These labels indicate whether the document is relevant (labeled as 1), not relevant (labeled as 0) or partially relevant (labeled as 2) to the respective query.

For each model, predictions were generated for a random sample of queries, and the associated documents were evaluated based on these predictions. The evaluation was conducted by comparing the predicted relevance against the actual labels.

After generating the predictions for the sampled queries, the results were saved in a CSV file, which includes the query text, document text, actual label, and predicted label for each query-document pair. These results were then used to assess the model's performance by comparing the qrels and the predicted values.



Serial No	Query	Document	Actual	GNN	GAE	GNN(Encoder-Decoder)
40287	coronavirus quarantine	During the 2003 outbreak of severe acute respiratory	0	2	2	2
76601	coronavirus diabetes	Diabetes is one of the most important comorbidities	1	0	0	1
79582	coronavirus biomarkers	The human coronaviruses have been shown to be a ma	0	2	2	2
17138	animal models of COVID-19	What is COVID-19, and what do we know so far about	1	2	2	1
49839	coronavirus outside body	Name of Virus: Coronavirus	2	1	0	2
17216	animal models of COVID-19	What is COVID-19, and what do we know so far about	2	2	2	2
14306	animal models of COVID-19	What is COVID-19, and what do we know so far about	2	2	2	2
74696	coronavirus hypertension	The most distinctive comorbidities of 32 non-survey	0	0	0	0
56786	coronavirus clinical trials	Background: Although a number of antiviral agents	0	2	2	0
33314	coronavirus social distancing impact	A novel Coronavirus pandemic emerged in December o	0	2	2	2
86897	coronavirus asymptomatic	Name of Virus: Coronavirus	2	1	1	2
98793	coronavirus vaccine candidates	Two novel coronaviruses have emerged in humans in	0	0	0	0
41903	how does coronavirus spread	Two recent studies provide initial insights into a	0	1	1	0

Figure 9: Comparison of Document Retrieval Models (GNN, GAE, and Encoder-Decoder) Against Actual Relevance for COVID-19 Queries.

The performance evaluation (Figure 10) of the models provide insightful information regarding their relative strengths and weaknesses in predicting the relevance of query-document pairs.

As shown in the comparison given in Figure 13, the model that combines Graph Neural Networks (GNN) with an Encoder-Decoder architecture outperforms the other models, achieving an accuracy of 51.46%. This is followed by the Graph Autoencoder (GAE), with an accuracy of 49.15%, and the basic GNN model, which attains an accuracy of 46.29%. These findings underscore the significant impact of integrating encoder-decoder frameworks and autoencoding approaches, which appear to significantly boost the models' ability to better capture the underlying relationships between queries and documents.

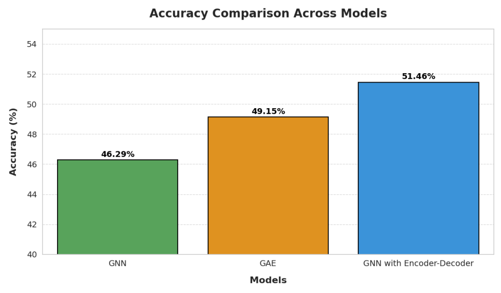


Figure 10: Comparison of accuracy across GNN, GAE, and GNN with Encoder-Decoder models.

6 CONCLUSION

Graph Neural Networks (GNNs) have become pivotal in machine learning, mainly designed to address graph-based operations. Using message-passing mechanisms, GNNs excel in capturing complex relationships within graphs, making them effective for applications like classification, link prediction and clustering. This work explores the fundamentals of GNNs, diverse models such as Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs) and their variants in domains like social networks and sciences. GNN supported by the GAE layers improved the accuracy of the classification process. Future research directions for GNNs include enhancing model scalability, interpretability, and assessment metrics, as well as exploring new applications in conventional graph algorithms. The capability of GNNs to revolutionize numerous fields is substantial promising great advancements in tackling real-world problems through research and development.

REFERENCES

Agarwal, C., Queen, O., Lakkaraju, H., and Zitnik, M. (2022). Evaluating explainability for graph neural networks.

Kavitha, K., Pranav, S., and Anil, A. (2023). Word sense disambiguation using supervised learning. In *2023 4th IEEE Global Conference for Advancement in Technology (GCAT)*, pages 1–6. IEEE.

Khan, I. Z., Sheikh, A. A., and Sinha, U. (2024). Graph neural network and ner-based text summarization.

Khemani, B., Patil, S., Kotecha, K., and Tanwar, S. (2024). A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, 11.

Li, J., Jian, Y., and Xiong, Y. (2024). Text classification model based on graph attention networks and adversarial training. *Applied Sciences*, 14:4906.

Li, P., Yang, Y., Pagnucco, M., and Song, Y. (2022). Explainability in graph neural networks: An experimental survey.

Menon, R. R., Kaartik, J., Karthik Nambiar, E., Tk, A. K., and Arun Kumar, S. (2020). Improving ranking in document based search systems. page 914 – 921.

Menon, R. R., Rahul, R., and Bhadrakrishnan, V. (2023). Graph auto encoders for content-based document recommendation system.

Namburu, S. S. G., Soman, K., Kumar, S. S., and Mohan, N. (2024). Effectiveness of gnn based approach for topic classification of telugu text. In *2023 4th International Conference on Intelligent Technologies (CONIT)*, pages 1–5. IEEE.

Rastakhiz, F., Davar, O., and Eftekhari, M. (2024). Beyond words: A heterogeneous graph representation of text via graph neural networks for classification. In *2024 20th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP)*, pages 1–7.

Reghu, L., Ashok, G., and Menon, R. R. (2024). Explainable ai for health care based retrieval system. volume 2, page 1915 – 1922.

Visweswaran, M., Mohan, J., Kumar, S. S., and Soman, K. (2024). Synergistic detection of multimodal fake news leveraging textgcn and vision transformer. *Procedia Computer Science*, 235:142–151.

Wang, K., Ding, Y., and Han, S. C. (2023). Graph neural networks for text classification: A survey.

Yuan, H., Yu, H., Gui, S., and Ji, S. (2023). Explainability in graph neural networks: A taxonomic survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5782–5799.

Zhang, Y., Yu, X., Cui, Z., Wu, S., Wen, Z., and Wang, L. (2020). Every document owns its structure: Inductive text classification via graph neural networks. *CoRR*, abs/2004.13826.

Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., and Sun, M. (2018). Graph neural networks: A review of methods and applications.