CiteGCN-LLM: Citation-Aware Research Papers Classification via Graph Convolutional Networks and Large Language Models

Anonymous ACL submission

Abstract

The rapid proliferation of scholarly publications has made the automated classification and recommendation of research articles essential for efficient scientific knowledge management. In this study, we propose CiteGCN-LLM, a unified citation-graph-based classification framework for Research Papers Classification (RPC) recommendations that synergistically integrates Graph Convolutional Networks (GCN) and Large Language Models (LLM). We introduce two types of citation-based graph constructions: **(1) RPCG-1**, which captures paper–word relationships enriched by cited-by papers, and **(2) RPCG-2**, which captures paper–author relationships along with their cited-by papers. To inject rich contextual semantics into the graph, we extract deep textual representations from paper abstracts using a pre-trained transformer-based LLM and fuse them with graph-based embeddings to ensure a tight alignment between textual meaning and structural citation cues. We further employ a Top-n labeling strategy using paper titles to personalize classification for recommendations to individual user preferences, and curate specialized citation datasets to support our experiments. Extensive evaluations on our citation datasets (e.g., arXiv, DBLP, Elsevier, and PubMed) demonstrate that CiteGCN-LLM significantly outperforms state-of-the-art baselines in terms of both classification accuracy and robustness. Our results demonstrate that integrating topological citation networks with deep semantic language understanding can significantly advance the development of intelligent academic search and recommendation systems.

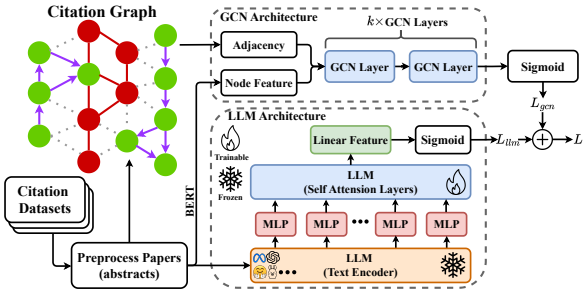


Figure 1: The illustration of CiteGCN-LLM for the RPC framework. A citation graph is constructed, and paper abstracts are encoded with a pretrained BERT to obtain node features. GCN layers capture structural information, while MLP and LLM components refine semantic features. Sigmoid classifiers are applied to both branches, and their outputs are combined for the final prediction.

1 Introduction

Across various fields, the rapid growth of scholarly articles has generated an overwhelming body of literature, making it harder for researchers to find the papers they need efficiently. Although today's scholars rely heavily on digital platforms, most of these systems fall short of providing effective paper classification and recommendations because of their limited ability to perform deep semantic analysis and to model relationships within academic texts Sugiyama and Kan (2010); Kanwal and Amjad (2024). For example, Google Scholar Lund et al. (2023) highlights highly cited papers by retrieving similar documents based on user queries and ranking results according to citation counts. Elsevier Sami et al. (2024) employs a hybrid recommendation engine that learns user preferences and paper features from activity logs and metadata through deep learning to detect shifts in paper popularity over time. Springer Nature Springer Nature (2024) utilizes NLP and deep-learning embeddings to analyze metadata and its proprietary ontology, helping to sug-

gest relevant research customized to individual interests. However, these methods still struggle to capture subtle semantic relationships in scholarly texts or accurately reflect users' personal preferences and the latest research trends. Notably, Google Scholar's dependence on citation counts means that highly cited papers appear at the top of search results, increasing their visibility, while recently published papers, which could be highly relevant, remain challenging to discover due to their initially low citation counts Perc (2014).

Citations contain vital structural information such as acknowledging prior work, indicating methodological lineage, or contrasting results that cannot be fully conveyed by standalone text models Tsai et al. (2023). A major limitation of purely text-based approaches, regardless of their complexity, is their inability to utilize the rich relational contexts among documents Cohan et al. (2020). Research papers are not isolated units; it is intricately connected through explicit citations, implicit co-authorship networks, shared methodologies, and thematic developments. By focusing only on a document's internal text, these approaches treat each research paper as an isolated source of information. It can understand the island's landscape and its content, but remains unaware of the surrounding ocean of academic discourse and the bridges (citations) that connect it to other scholarly islands. This results in a fragmented understanding, where a paper's true scope, influence, and role within a larger academic conversation are not fully captured.

Researchers often have to manually sift through ever-growing collections of data, an exhausting and inefficient task that wastes valuable time, reduces research productivity, and risks missing important interdisciplinary insights. Traditional categorization methods struggle to capture the complex, cross-domain connections that define modern science, causing researchers to either overlook key discoveries or spend endless hours sorting unrelated work. To address these limitations, we propose a CiteGCN-LLM framework. This end-to-end framework combines a citation-aware GCN with a transformer-based language understanding model for multi-label paper classification. We first build two heterogeneous graphs (1) paper–word (using citations, TF–IDF, and co-occurrence) and (2) paper–author (using citations and co-authorship)—and seed each node with its pooled paper (abstract) embedding. A multi-layer GCN then propagates these embeddings across the graph to capture nuanced citation intents. Meanwhile, a parallel transformer head directly produces logits from the raw text to capture semantic relationships. By training both modules jointly under a unified binary cross-entropy loss, we allow structural citation cues and textual semantics to reinforce one another. During inference, the outputs of the GCN and LLM are combined to generate the final label predictions. This integrated design results in richer representations and achieves state-of-the-art accuracy and robustness on our citation datasets.

Our main contributions are summarized as follows:

- We curate four citation datasets (arXiv, DBLP, Elsevier, PubMed) enriched with comprehensive paper metadata, abstracts, and multi-label topics. These datasets serve as a crucial and unified resource for evaluating both graph and text-based methods in the context of citation analysis.

- We propose two novel heterogeneous graph types: (1) *RPCG-1*, a paper-word graph incorporating citation counts, TF-IDF weights, and positive PMI for term relations; and (2) *RPCG-2*, a paper-author graph capturing directed citations, co-authorship, and author-paper affiliations. These constructions are designed to capture diverse structural signals crucial for discerning citation intents.

- We design CiteGCN-LLM, a unified framework that combines transformer-pooled text embeddings with multi-layer GCN propagation over citation graphs, along with a concurrent transformer-based classifier. By jointly minimizing GCN and LLM cross-entropy losses and employing a Top-K labeling scheme, it provides interpretable multi-label targets for RPC.

- Extensive qualitative and quantitative evaluations across multiple citation datasets confirm that CiteGCN-LLM

provides high performance in classification for personalized recommendations, with enhanced semantic coherence for research paper retrieval.

2 Related Work

Recent work on scholarly text classification can be broadly grouped into three directions. First, several studies combine pretrained language models with citation graphs to improve document classification and recommendation, including BERT augmented with citation encoders Yasunaga et al. (2023) and heterogeneous Graph Neural Networks (GNN) that jointly model textual and citation information Wang et al. (2023). Second, graph-based transformer models have shown strong performance on scholarly networks, including Graph-BERT for citation modeling Zhou et al. (2023) and graph transformers designed for scientific document classification Dai et al. (2024). Third, hybrid GCN–Transformer architectures have been explored for robust document labeling and recommendation Jia et al. (2024); Chen et al. (2025).

Distinct from these approaches, we propose a CiteGCN–LLM framework that separately models citation structure and textual semantics and jointly optimizes both components under a unified multi-label objective. This design avoids complex graph transformers and parameter-heavy fusion, effectively integrating structural and semantic information.

3 Preliminaries

3.1 Text-Attributed Graphs

A text-attributed graph enriches nodes (e.g., papers, authors, keywords) with textual embeddings (e.g., TF–IDF or transformer-pooled) and edges that encode relations such as citations, co-authorship, or term co-occurrence. This structure enables joint modeling of semantic content and graph topology Yao et al. (2019).

3.2 Graph Convolutional Networks

GCN learn node representations by iteratively aggregating and transforming features of neighboring nodes through the normalized adjacency matrix and nonlinear functions. Multiple layers enable information to flow across the graph, capturing both local structure and node attributes Kipf and Welling (2017).

3.3 Large Language Models

LLMs (e.g., BERT, GPT) use multi-head self-attention and feed-forward layers to produce contextual token embeddings. These models understand long-range dependencies in text and act as powerful feature extractors or classifiers for downstream tasks Vaswani et al. (2017); Devlin et al. (2019).

4 Approach

Figure 1 illustrates the CiteGCN–LLM framework. The process commences with preprocessing the citation dataset to extract titles and abstracts and to construct a heterogeneous paper–word–citation graph encoding citation links, TF–IDF-based paper–word associations, and word co-occurrences. The Citation-Aware GCN utilizes a multi-layer graph convolutional network across the comprehensive graph, propagating structural information via stacked convolutions to produce citation-informed logits for paper nodes. Concurrently, the Textual LLM processes tokenized abstracts using a pretrained language model with self-attention mechanisms and a streamlined classification head, yielding semantic logits. The GCN and LLM are jointly trained by minimizing the sum of their binary cross-entropy losses, and, during inference, their logits are integrated to combine structural and semantic information.

4.1 Graph Representation

4.1.1 Notation

We represent the citation-aware text corpus as an attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X)$, where the node set $\mathcal{V} = \mathcal{P} \cup \mathcal{W}$ consists of paper nodes $\mathcal{P} = \{p_1, \ldots, p_{N_p}\}$ and word nodes $\mathcal{W} = \{w_1, \ldots, w_M\}$, with $|\mathcal{V}| = N_p + M$. The edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ includes document–word co-occurrence edges, word–word co-occurrence edges, and citation edges between papers. A directed citation edge $(p_i, p_j) \in \mathcal{E}$ indicates that paper p_j cites paper p_i. Each node $v \in \mathcal{V}$ is associated with a feature vector $x_v \in \mathbb{R}^d$, forming the feature matrix $X \in \mathbb{R}^{|\mathcal{V}| \times d}$.

4.1.2 RPCG-1 (Paper-Word-Citation)

In our initial citation graph construction, we create a single heterogeneous network compris-

ing paper and word nodes, enabling the model to utilize both structural citation data and lexical semantics simultaneously. Edges between paper nodes represent direct citation counts, maintaining the direction and strength of scholarly influence. Word–word edges reflect semantic similarity through positive pointwise mutual information (PMI), ensuring only strongly co-occurring terms are connected. Paper–word edges are weighted with TF–IDF scores, anchoring each paper in its textual content and allowing graph convolution to transmit document-specific term importance. Equation (1) defines RPCG-1.

where, $\mathrm{PMI}(w_i, w_j)$ is the pointwise mutual information between words w_i and w_j, included only if positive to reflect strong semantic association. $\mathrm{TF-IDF}_{i,j}$ is the TF–IDF weight of term w_j in paper p_i, connecting textual content to the graph. $(C_{pp})_{i,j}$ is the number of citations from paper p_j to paper p_i, preserving the directed citation structure.

$$A_{ij} = \begin{cases} \mathrm{PMI}(i,j), & i,j \in \mathcal{W},\ \mathrm{PMI}(i,j) > 0 \\ \mathrm{TF-IDF}_{ij}, & i \in \mathcal{P},\ j \in \mathcal{W} \\ (C_{pp})_{i,j}, & i,j \in \mathcal{P},\ p_i \text{ is cited by } p_j \\ 1, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

$$A_{ij} = \begin{cases} \mathrm{PMI}(a_i, a_j), & i,j \in \mathcal{A},\ \mathrm{PMI}(a_i,a_j) > 0 \\ (A_{ap})_{i,j}, & i \in \mathcal{P},\ j \in \mathcal{A},\ a_j \in p_i \\ (C_{pp})_{i,j}, & i,j \in \mathcal{P},\ p_i \text{ is cited by } p_j \\ 1, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

4.1.3 RPCG-2 (Paper-Author-Citation)

In the RPCG-2 schema, we represent papers and authors within a single, unified graph whose edges carry detailed, diverse information. Citation connections between papers are weighted by the number of times one paper cites another, enabling the network to capture direct scholarly influence. Between author nodes, we encode co-authorship frequency so that authors with more shared publications have stronger connections, thus modeling the strength of their research collaborations. We also include bipartite connections between each author and the papers they have authored, allowing textual and structural information to flow freely between these two data types. Equation (2) defines RPCG-2. The PMI value of an author pair i, j is computed as:

$$\mathrm{PMI}(a_i, a_{i'}) = \log \frac{\Pr(a_i, a_{i'})}{\Pr(a_i)\ \Pr(a_{i'})} \tag{3}$$

$$\Pr(a_i, a_{i'}) = \frac{|P_i \cap P_{i'}|}{N_p} \tag{4}$$

$$\Pr(a_i) = \frac{|P_i|}{N_p} \tag{5}$$

$$\Pr(a_{i'}) = \frac{|P_{i'}|}{N_p} \tag{6}$$

where, $|P_i|$ is the number of papers authored by a_i, $|P_i \cap P_{i'}|$ is the number of papers co-authored by both a_i and $a_{i'}$, and N_p is the total number of papers in the corpus. A positive PMI value indicates that the two authors collaborate more frequently than would be expected by chance (i.e. a strong co-authorship signal). In contrast, a negative PMI value suggests no excess collaboration beyond random pairing. $(C_{pp})_{i,j}$ represents the number of times paper p_j cites paper p_i, maintaining the directed citation structure among papers if co-authors are related to this paper. Finally, the bipartite author–paper adjacency A_{ap} is a binary $N_a \times N_p$ matrix, where $(A_{ap})_{i,j} = 1 \iff a_i$ is an author contributed to paper p_j, connecting each author node directly to their published works.

4.1.4 Node Feature Representation

We construct the initial node-feature matrix $X^{(0)} \in \mathbb{R}^{(M+N_p)\times d}$, where the first M rows correspond to word nodes and the next N_p rows to paper nodes, and X is the all node feature matrix.

- **Word nodes,** w_i $(i = 1, \ldots, M)$: We assign each word its pretrained embedding as follows:

$$X^{(0)}_{w_i,:} = e(w_i) \in \mathbb{R}^d \tag{7}$$

where, $e(w_i)$ is the BERT embedding of w_i.

- **Paper nodes,** p_j $(j = 1, \ldots, N_p)$: We aggregate the embeddings of words that appear in paper p_j using the word–paper biadjacency matrix $A_{wp} \in \{0,1\}^{M \times N_p}$ as defined:

$$X^{(0)}_{p_j,:} = \frac{1}{\sum_{i=1}^{M} (A_{wp})_{i,j}} \sum_{i=1}^{M} (A_{wp})_{i,j}\, e(w_i) \;\in\; \mathbb{R}^d \tag{8}$$

- **Author nodes,** a_i: We initialize the joint author–paper feature matrix $X^{(0)} \in \mathbb{R}^{(N_a+N_p)\times d}$ by assigning each node a d-dimensional embedding. In particular, for each author node a_i, we aggregate the BERT-pooled paper embeddings of all papers they have authored:

$$X^{(0)}_{a_i,:} = \frac{1}{|P_i|} \sum_{p_j \in P_i} h_j^{(\mathrm{BERT})}, \quad P_i = \{\, p_j \mid (A_{\mathrm{ap}})_{i,j} = 1 \}$$

(9)

where, P_i is the set of papers authored by a_i, $h_j^{(\mathrm{BERT})} \in \mathbb{R}^d$ is the pooled abstract embedding for paper p_j, and A_{ap} represents the author–paper bi-adjacency matrix.

4.2 Data Labeling

We employ a Top-K unique word-based labeling strategy to construct multi-label supervision for RPC using research paper titles. Given P papers and a vocabulary of size V, we first build a TF–IDF document–term matrix as follows:

$$T \in \mathbb{R}^{P \times V} \tag{10}$$

where, $T_{j,i}$ denotes the TF–IDF weight of term w_i in paper p_j.

To identify globally salient unique words, we aggregate TF–IDF scores over the corpus:

$$s_i = \textstyle\sum_{j=1}^{P} T_{j,i}, \qquad \{\pi(1), \ldots, \pi(K)\} = \arg\text{top-K}_i \; s_i \tag{11}$$

Using the selected Top-K unique words $\{w_{\pi(1)}, \ldots, w_{\pi(K)}\}$, each paper p_i is assigned a binary label vector $Y_i \in \{0,1\}^K$ defined as:

$$Y_{ik} = \mathbb{I}\big(w_{\pi(k)} \in p_i\big) \tag{12}$$

where, $\mathbb{I}(\cdot)$ is the indicator function. Stacking all Y_i yields the label matrix $Y \in \{0,1\}^{P \times K}$, which is used as multi-label supervision for downstream training.

4.3 GCN-LLM Model

4.3.1 GCN Forward Pass

We model the corpus as a heterogeneous, citation-aware graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where nodes represent papers, words, and authors, and edges capture citation relations, TF–IDF-based paper–word and paper–author associations, and word and author co-occurrences, to implement the same process for RPCG-1 and RPCG-2. Each node is initialized with a feature vector derived from textual representations using the BERT model. Let $A \in \mathbb{R}^{N \times N}$ denote the adjacency matrix of $\mathcal{G}$. To enable stable message passing, we add self-loops and apply symmetric normalization, $\tilde{A} = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}}$, where $Dii = \sum_j (Aij + I_{ij})$ is degree matrix and I is the identity matrix. The normalized adjacency $\tilde{A}$ ensures degree-balanced aggregation of information from neighboring nodes. Given the initial node feature matrix $X \in \mathbb{R}^{N \times d}$, an L-layer GCN iteratively computes hidden representations by propagating and transforming information over the graph structure.

First layer.

$$H^{(1)} = \sigma\Big(\tilde{A} X W^{(1)} + b^{(1)}\Big) \tag{13}$$

where, $W^{(1)} \in \mathbb{R}^{d \times h}$, $b^{(1)} \in \mathbb{R}^h$, and $\sigma(\cdot)$ denotes the ReLU activation.

Intermediate layers. $(2 \le \ell \le L-1)$**:**

$$H^{(\ell)} = \sigma\Big(\tilde{A} H^{(\ell-1)} W^{(\ell)} + b^{(\ell)}\Big) \tag{14}$$

where, $H^{(\ell-1)} \in \mathbb{R}^{N \times h}$ is the output of the previous layer, $W^{(\ell)} \in \mathbb{R}^{h \times h}$ and $b^{(\ell)} \in \mathbb{R}^{1 \times h}$ are layer-specific parameters.

Final layer. The last layer produces node-wise logits without nonlinearity:

$$Z_{\mathrm{gcn}} = H^{(L-1)} W^{(L)} \tag{15}$$

where, $W^{(L)} \in \mathbb{R}^{h \times K}$ projects each node representation into K output dimensions corresponding to the Top-K labels. For training, logits for paper nodes are selected and optimized using binary cross-entropy with the TF–IDF-based multi-label targets.

4.3.2 LLM Forward Pass

For each paper $p_i \in \mathcal{I}_{\text{train}}$, we construct tokenized inputs $I_{\text{train}} \in \mathbb{Z}^{|\mathcal{I}_{\text{train}}| \times S}$ and corresponding attention masks $M_{\text{train}} \in \{0,1\}^{|\mathcal{I}_{\text{train}}| \times S}$, where S denotes the maximum sequence length. These inputs are fed into a pretrained transformer-based classifier, yielding multi-label logits:

$$Z_{\text{llm}}^{\text{train}} = \text{LLM}_{\theta_{\text{llm}}}(I_{\text{train}}, M_{\text{train}}) \in \mathbb{R}^{|\mathcal{I}_{\text{train}}| \times K} \tag{16}$$

Each row of $\text{Z}_{\text{llm}}^{\text{train}}$ represents a K-dimensional vector of unnormalized scores corresponding to the Top-K labels derived from TF–IDF-based data labeling. During training, these logits are directly optimized using binary cross-entropy with logits against the ground-truth label matrix Y_{train}.

4.3.3 GCN–LLM Model Loss

We introduce a concatenated loss for our model that integrates a GCN and an LLM. Let $\mathcal{I}_{\text{train}}$ denote the set of training paper indices. Let $Y \in \{0,1\}^{|\mathcal{I}_{\text{train}}| \times K}$ be the multi-hot label matrix used in training. The GCN produces logits $Z_{\text{gcn}} = f_\theta(X, A) \in \mathbb{R}^{|\mathcal{V}| \times K}$ for all nodes, and we select $Z_{\text{gcn}}^{\text{train}} = Z_{\text{gcn}}[\mathcal{I}_{\text{train}}]$. The LLM classifier (BERT) produces logits $Z_{\text{llm}}^{\text{train}} = g_\phi(\texttt{input\_ids}, \texttt{attention\_mask}) \in \mathbb{R}^{|\mathcal{I}_{\text{train}}| \times K}$.

We adopt the binary cross-entropy with logits loss defined as:

$$\mathcal{L}_{\text{BCELogits}}(Z, Y) = \frac{1}{|\mathcal{I}_{\text{train}}| K} \sum_{i=1}^{|\mathcal{I}_{\text{train}}|} \sum_{k=1}^{K} \Big[\max(Z_{ik}, 0) - Z_{ik} Y_{ik} + \log\big(1 + \exp(-|Z_{ik}|)\big) \Big] \tag{17}$$

where, Z_{ik} denotes the predicted logit for sample i and label k, $Y_{ik} \in \{0,1\}$ is the corresponding binary target, $|\mathcal{I}_{\text{train}}|$ is the number of training samples, and K is the number of labels. This formulation is numerically stable and equivalent to binary cross-entropy applied to $\sigma(Z_{ik})$.

The GCN and LLM losses are defined as:

$$\begin{aligned} \mathcal{L}_{\text{gcn}}^{\text{train}} &= \mathcal{L}_{\text{BCELogits}}\big(Z_{\text{gcn}}^{\text{train}}, Y\big), \\ \mathcal{L}_{\text{llm}}^{\text{train}} &= \mathcal{L}_{\text{BCELogits}}\big(Z_{\text{llm}}^{\text{train}}, Y\big) \end{aligned} \tag{18}$$

Following our implementation, we jointly optimize both components using the unweighted sum as follows:

$$\mathcal{L}^{\text{train}} = \mathcal{L}_{\text{gcn}}^{\text{train}} + \mathcal{L}_{\text{llm}}^{\text{train}} \tag{19}$$

Backpropagation is performed once on $\mathcal{L}^{\text{train}}$ and we update both parameter sets θ and ϕ in the same iteration. (See Appendix B for additional details.)

5 Experimental Settings

5.1 Datasets and Preprocess

Experiments are conducted using four citation datasets: (1) arXiv, (2) DBLP, (3) Elsevier, and (4) PubMed. We initially retrieved metadata for each dataset's papers—including titles, authors, abstracts, keywords, and publication years using the HuggingFace[1] platform. To enrich these collections with citation contexts, we then used ScraperAPI[2] to query Google Scholar for each paper's "cited-by" list. Specifically, for every title in our Hugging Face datasets, we retrieved data on the Top-20 citing papers (including titles, authors, abstracts, and keywords) published between 2023 and 2025; if fewer than 20 citations are available, we kept all available cited-by papers. This process results in four specialized citation graphs for research-paper classification that incorporate current scholarly influence information. Table 1 compares the source datasets and reports their preprocessing statistics.

In preprocessing, we clean and structure raw text using NLTK[3] by (1) removing punctuation, including URLs and symbols, (2) removing stopwords, (3) lemmatizing tokens to their base forms, and (4) converting terms to nouns to standardize and formalize the vocabulary. This process retains only meaningful, normalized concepts represented as concrete noun forms for downstream modeling.

	Datasets			
	arXiv	**DBLP**	**Elsevier**	**PubMed**
# Total Papers	2,744	2,096	2,796	2,116
# Cite Papers	1,744	1,577	1,224	1,058
# Vocabulary	1,368	1,020	1,257	1,240
# Authors	2,744	1,787	1,058	1,374
# Node (paper–word)	4,112	3,116	4,053	3,356
# Node (paper–author)	5,488	3,883	3,854	3,490
# Edge (paper–word)	185,526	117,298	171,227	184,013
# Edge (paper–author)	161,113	182,690	172,887	153,657
# Average Length	278.65	321.90	370.72	279.93

Table 1: Summary of the datasets used in our experiments.

[1] https://huggingface.co/datasets
[2] https://dashboard.scraperapi.com
[3] https://www.nltk.org/api/nltk.html

5.2 Model Selection

We use a GCN Yao et al. (2019) to capture the detailed relational structure of the citation graph, and we fine-tune a pre-trained LLM (`bert-base-uncased`) Nihalani and Shah (2024) to extract deep semantic embeddings from paper abstracts, thereby combining topological and textual information in our classification framework. (See Appendix C for additional details.)

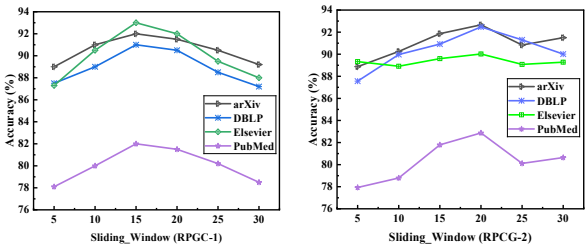
6 Performance Evaluations

6.1 Test Performance

Table 2 presents several key findings. First, baseline GNN such as GCN, GAT, GraphSAGE, and GIN plateau in the mid-80% range for both accuracy and F1 scores. In contrast, LLaMA-enhanced variants (e.g., $\text{GCN}_{\text{Llama2}}$, $\text{GIN}_{\text{Llama3}}$) consistently improve performance by 2–4 percentage points across all four datasets, demonstrating the standalone strength of deep language embeddings. Second, our CiteGCN-LLM models trained jointly on RPCG-1 (paper–word) and RPCG-2 (paper–author) heterogeneous graphs alongside transformer-pooled text features achieve the highest results, surpassing 93% on arXiv and approximately 92% on DBLP. Notably, RPCG-2's author-centric edges yield the largest gains, particularly on the socially connected DBLP corpus. Finally, average gains of about 3–3.7% in both accuracy and F1 confirm that CiteGCN_LLM in graph and text-based models significantly enhances representation quality. Overall, these results demonstrate that integrating structural citation relations with advanced language understanding in an efficient CiteGCN_LLM provides a powerful, practical approach to next-generation scholarly classification and recommendation.

6.2 Parameter Sensitivity

Figure 2(a) illustrates that for RPCG-1, classification accuracy across all four corpora consistently improves as the sliding window expands from 5 to 15, reaching a peak around the 15-20 range, before gradually declining with a window size of 30. This observation confirms that moderate context sizes are most effective at capturing informative co-occurrence patterns: overly small windows overlook critical relationships, whereas large windows introduce unnecessary noise. In contrast, Figure 2(b) for RPCG-2 shows a slightly delayed peak (around window size 20), suggesting that author-based edges benefit from a broader context before performance starts to diminish. Collectively, these trends highlight the crucial role of sliding-window granularity in balancing locality and globality in graph construction, and underscore our selection of an optimal window size for subsequent citation-graph classification.



(a) (b)

Figure 2: Impact of sliding window size on the classification performance of RPCG-1 and RPCG-2.

6.3 Learning Rate Sensitivity

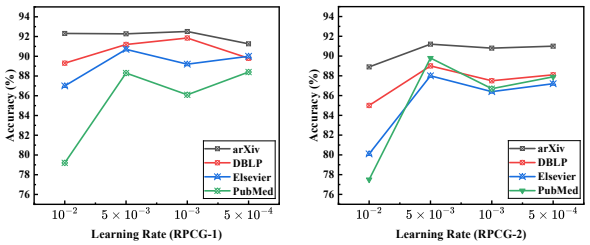
In our experiments, we observe that Figure 3(a) demonstrates significant improvement with a moderate learning rate. The accuracy on arXiv, DBLP, Elsevier, and PubMed shows a steep increase as the learning rate rises from 10^{-2} to 5×10^{-3}, peaks at 5×10^{-3}, and subsequently declines when the rate exceeds 10^{-3}. This trend indicates that a learning rate that is too small hinders convergence, while one that is too large leads to oscillations. In contrast, Figure 3(b), which utilizes paper–author and author–author edges, achieves optimal performance at an even lower learning rate of 10^{-3}, reflecting its heightened sensitivity to step size. These findings emphasize the critical need for optimized learning-rate tuning corresponding to each graph-construction strategy to ensure stable and efficient optimization in citation-graph classification.

6.4 Ablation Study

Table 3 presents two significant findings regarding graph construction for citation graph-based classification. First, enhancing both RPCG-1 and RPCG-2 with citation edges (C_{p2p}) results in substantial improvements in accuracy and F1-scores (for instance, RPCG-1 on arXiv increases from 89.91% to 92.97% accuracy).

Models	arXiv		DBLP		Elsevier		PubMed	
	Acc.↑	F1↑	Acc.↑	F1↑	Acc.↑	F1↑	Acc.↑	F1↑
GCN Wu et al. (2019)	86.53(0.92)	85.66(0.78)	86.12(0.93)	85.64(0.82)	86.11(0.35)	85.39(0.40)	71.74(0.21)	71.04(0.37)
GAT Veličković et al. (2018)	86.12(0.95)	85.05(0.88)	85.49(0.76)	84.89(0.71)	84.37(0.28)	84.79(0.31)	73.66(0.33)	72.44(0.19)
GraphSAGE Hamilton (2017)	87.08(0.85)	85.96(0.73)	87.69(0.92)	87.38(0.68)	83.52(0.45)	83.91(0.50)	71.19(0.26)	70.87(0.45)
GIN Xu et al. (2019)	86.60(0.91)	85.37(0.74)	85.84(0.92)	85.31(0.68)	85.76(0.22)	85.59(0.27)	71.62(0.47)	71.13(0.33)
BernNet He et al. (2021)	88.52(0.95)	87.96(0.85)	88.48(0.41)	87.52(0.79)	84.93(0.51)	84.28(0.48)	72.97(0.17)	71.77(0.22)
FAGCN Ha et al. (2021)	88.55(1.36)	87.92(0.65)	89.08(0.54)	88.72(0.53)	87.81(0.30)	87.20(0.35)	73.11(0.20)	72.14(0.38)
GCNII Chen et al. (2020)	88.93(1.37)	87.58(0.71)	89.80(0.30)	88.96(0.62)	86.37(0.40)	86.06(0.42)	72.74(0.00)	72.22(0.44)
RevGAT Li et al. (2021)	89.11(1.00)	87.65(0.58)	88.50(0.05)	87.12(0.53)	85.29(0.25)	85.61(0.29)	74.02(0.18)	73.56(0.29)
ACM-GCN Lu et al. (2022)	89.75(1.16)	88.94(0.54)	90.96(0.62)	89.77(0.51)	88.01(0.55)	87.46(0.50)	73.62(0.14)	72.96(0.26)
Graphormer Ying et al. (2024)	80.41(0.30)	79.98(0.56)	88.24(1.50)	87.52(0.71)	83.69(0.33)	83.81(0.37)	72.63(0.13)	71.58(0.42)
$GCN_{(Llama2)}$	90.59(0.71)	89.62(0.66)	88.97(0.82)	88.56(0.71)	85.57(0.48)	85.14(0.45)	73.87(0.22)	73.87(0.61)
$GCN_{(Llama3)}$	90.77(0.28)	90.35(0.37)	89.76(0.47)	89.65(0.31)	84.25(0.26)	84.08(0.29)	74.68(0.45)	74.29(0.32)
$GAT_{(Llama2)}$	90.33(0.67)	89.72(0.59)	87.93(0.28)	87.42(0.54)	83.93(0.44)	83.71(0.47)	74.92(0.14)	74.48(0.10)
$GAT_{(Llama3)}$	91.51(0.35)	91.45(0.58)	88.31(0.28)	87.93(0.63)	84.25(0.26)	84.49(0.29)	75.71(0.41)	75.06(0.32)
$GraphSAGE_{(Llama2)}$	90.22(0.77)	89.89(0.19)	89.96(0.50)	89.67(0.34)	85.10(0.32)	85.38(0.28)	72.53(0.61)	72.42(0.49)
$GraphSAGE_{(Llama3)}$	91.70(0.57)	91.08(0.62)	90.14(0.56)	89.96(0.30)	83.96(0.44)	83.72(0.47)	73.36(0.38)	73.25(0.32)
$GIN_{(Llama2)}$	90.26(0.67)	89.20(0.48)	87.73(0.82)	87.51(0.30)	88.33(0.36)	87.82(0.39)	73.71(0.25)	73.42(0.19)
$GIN_{(Llama3)}$	91.33(0.28)	91.05(0.53)	89.22(0.30)	89.22(0.14)	88.49(0.38)	87.08(0.41)	75.64(0.46)	75.28(0.39)
CiteGCN-LLM$_{(RPCG-1)}$	**93.58 (0.45)**	**93.17 (0.42)**	**91.15 (0.40)**	**91.82 (0.39)**	**89.75 (0.28)**	**89.29 (0.30)**	**82.95 (0.35)**	**81.25 (0.34)**
CiteGCN-LLM$_{(RPCG-2)}$	**93.10 (0.50)**	**92.51 (0.45)**	**91.73 (0.48)**	**90.57 (0.44)**	**89.42 (0.30)**	**88.75 (0.32)**	**81.97 (0.40)**	**80.58 (0.38)**
Avg. Gains$_{(RPCG-1)}$	2.02%	2.29%	1.12%	2.05%	1.42%	2.51%	3.56%	3.70%
Avg. Gains$_{(RPCG-2)}$	1.53%	1.60%	1.76%	0.68%	1.05%	1.96%	3.68%	2.82%

Table 2: Node classification performance of various citation-graph models on the citation datasets. The highest scores are shown in **bold** in orange (RPCG-1) and magenta (RPCG-2), and the second-best scores in gray. "Avg. Gains" reports the average improvement achieved by each model (RPCG-1 and RPCG-2) through the CiteGCN-LLM model.



(a) (b)

Figure 3: Performance of learning rate impact based on RPCG-1 and RPCG-2 models.

Method	Graph Construction	arXiv		DBLP	
		Acc	F1	Acc	F1
RPCG-1	$(p2w + w2w)$	89.91	89.01	88.29	87.81
	$(p2w + w2w + C_{p2p})$	**92.97**	**91.41**	**91.99**	**91.33**
RPCG-2	$(p2a + a2a)$	89.57	88.60	88.58	88.13
	$(p2a + a2a + C_{p2p})$	**92.73**	**91.01**	**91.67**	**91.06**

Table 3: Performance of RPCG variants on the arXiv and DBLP datasets.

This confirms that explicit citation connections capture essential inter-paper semantics that extend beyond mere lexical or authorship cues. Second, even in the absence of citation edges, RPCG-2's author-centric topology (comprising paper–author and author–author relationships) slightly outperforms RPCG-1's text-centric design on both arXiv and DBLP. This indicates that social co-authorship patterns provide a complementary relational structure. Collectively, these insights highlight the importance of a heterogeneous graph integrating textual and citation relations to achieve the most robust classification performance. (See Appendix D for additional details.)

7 Conclusion

In this study, we introduced CiteGCN-LLM, a unified framework that combines heterogeneous citation graphs with transformer-pooled text embeddings, significantly improving RPC compared to methods that rely on text and graph data. By building both RPCG-1 (paper–word-citation) and RPCG-2 (paper–author-citation) graphs and jointly training GCN and LLM models, a personalized Top-K labeling strategy further boosts interpretability and relevance aligned with user preferences. Our framework effectively captures complementary semantic and topological signals, leading to notable improvements in accuracy and robustness across four citation datasets. However, its dependence on static snapshots of citations and co-authorship can limit its ability to respond to emerging research trends and cross-domain connections. Future work will focus on developing dynamic, temporally updated graphs and multimodal embeddings (e.g., full-text and figures) to reflect the evolving research landscape better and improve personalization in recommendations through adaptive labeling schemes.

References

Jian Chen, Jun Dong, and Lei Sun. 2025. Llm4rec: Large language models for scientific paper recommendation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1012–1023.

Ming Chen, Zhe Li, Yaliang Li, Yongbin Li, Xiaodan Liang, and Ming Zhang. 2020. Simple and deep graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Arman Cohan, Iz Beltagy, and Doug Downey. 2020. Scicite: A citation intent classification dataset for scientific document processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7115–7126. Association for Computational Linguistics.

Ziheng Dai, Pengcheng Li, and Lingpeng Kong. 2024. Graph transformer for scientific document classification. In *International Conference on Learning Representations (ICLR)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the ACL (NAACL)*, pages 4171–4186.

Seonghyeon Ha, Sujeong Lee, and Hyunwoo J. Kim. 2021. Fagcn: Frequency adaptive graph convolutional networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(9):4009–4021.

William L. Hamilton. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1024–1034.

Sheng He, Qitao Sun, and Philip S. Yu. 2021. Bernnet: Graph convolutional network with node-wise differentiated propagation. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, pages 13529–13537.

Yuanjun Jia, Fei Huang, and Wei Zhao. 2024. Hybrid gcn-transformer for document classification. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Tayyaba Kanwal and Taha Amjad. 2024. Research paper recommendation system based on multiple features from citation network. *Scientometrics*, 129(7):5493–5531.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Xiaofan Li, Peng Cui, and Wenwu Zhu. 2021. Reversible graph attention networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1019–1028.

Ying Lu, Tao Zhang, and Heng Ji. 2022. Acm-gcn: Adaptive citation modeling with gcn for scientific paper recommendation. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2463–2472.

Brian D. Lund, Thomas Wang, N. R. Mannuru, Bao Nie, Sanjib Shimray, and Zhen Wang. 2023. Chatgpt and a new academic reality: Artificial intelligence-written research papers and the ethics of the large language models in scholarly publishing. *Journal of the Association for Information Science and Technology*, 74(5):570–581.

Rahul Nihalani and Kushal Shah. 2024. Enhancing grammatical error detection using bert with cleaned lang-8 dataset. *arXiv preprint arXiv:2411.15523*. Fine-tuned `bert-base-uncased` achieves 0.91 F1 on GED.

Matjaž Perc. 2014. The matthew effect in empirical data. *Journal of the Royal Society Interface*, 11(98):20140378.

Ahmed Sami, Walid E. Adrousy, Salah Sarhan, and Sherif Elmougy. 2024. A deep learning based hybrid recommendation model for internet users. *Scientific Reports*, 14:29390.

Springer Nature. 2024. Find the right journal for your manuscript. Springer Nature Support. Online; accessed 2024-09-30.

Koichi Sugiyama and Min-Yen Kan. 2010. Scholarly paper recommendation via user's recent research interests. In *Proceedings of the 10th ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 29–38. ACM/IEEE.

Hao Tsai, Andrew Yen, Hsiu Huang, and Hsin-Hsi Chen. 2023. Citation intent classification and its supporting evidence extraction for citation graph construction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2472–2481, Birmingham, United Kingdom.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Xinyu Wang, Wentao Li, and Jie Liu. 2023. Citation-aware graph neural networks. In *Proceedings of The Web Conference (WWW)*, pages 567–578.

Fenglei Wu, Tianyi Zhang, Xingquan Zhu, and Jianping Yin. 2019. Simplifying graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

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

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7370–7377.

Michihiro Yasunaga, Yuxiang Peng, and SP Chang. 2023. Graph-based citation recommendation with bert. In *Proceedings of the 61st Annual Meeting of the ACL*, pages 1234–1248.

Raymond Ying, Jiaxuan You, Christopher Morris, Xiang Ren, and William L. Hamilton. 2024. Graphormer: A transformer-based framework for graph representation learning. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 22100–22115.

Jing Zhou, Petar Veličković, Xia He, and Wenlin Hu. 2023. Graph-bert: Only attention is needed for citation networks. In *Proceedings of The Web Conference (WWW)*, pages 1596–1606.

A Appendix

The supplementary material presents the following sections to strengthen the main manuscript:

- Section B provides more details about our approach, including the GCN–LLM training process, computational and memory complexity with analysis, and Mathematical Details and Proofs.
- Section C provides more details on the experimental environment, hyperparameter configurations, and dataset distribution.
- Section D provides more in-depth experimental results, discusses model limitations, and provides a discussion.
- Source Code Repositories: https://anonymous.4open.science/r/ACL-26-CiteGCN-LLM-5166/

B Approach

In this section, we provide further explanation of the GCN–LLM training process, analysis of computational and memory complexity, and mathematical details and proofs, all of which are essential for our study.

B.1 Joint GCN-LLM Training Process

Algorithm 1 presents the joint optimization procedure for the citation-aware GCN and the textual LLM under an early-stopping criterion. At each epoch, the gradients of both parameter sets θ_{gcn} and θ_{llm} are first reset. The GCN then performs a full-graph forward pass on (A, X) to produce node-wise logits Z_{gcn}, from which only entries corresponding to the training paper indices $\mathcal{I}_{\text{train}}$ are selected for supervision. This design allows message passing to exploit the entire citation graph while restricting loss computation to labeled nodes. In parallel, the LLM processes only the tokenized inputs of the training subset $(I_{\text{train}}, M_{\text{train}})$ to generate semantic logits $Z_{\text{llm}}^{\text{train}}$ with the same label dimensionality. The two branches are coupled through a unified multi-label objective, where the training loss $\mathcal{L}_{\text{train}}$ is defined as the sum of two BCEWithLogits terms evaluated against the shared multi-hot targets Y_{train}. This formulation implicitly applies an element-wise sigmoid and supports independent multi-label predictions. A single backpropagation step on $\mathcal{L}_{\text{train}}$ updates both parameter sets simultaneously, aligning structural and semantic representations. After each update, validation is performed by computing $\mathcal{L}_{\text{val}}$ from GCN and LLM logits restricted to the validation indices $\mathcal{I}_{\text{val}}$. Whenever $\mathcal{L}_{\text{val}}$ improves, the current parameters are saved as $(\theta_{\text{gcn}}^{\star}, \theta_{\text{llm}}^{\star})$ and the patience counter is reset; otherwise, the counter is incremented and training terminates once it reaches p, ensuring early stopping and selection of the best-performing model.

B.2 Computational Complexity

Graph Convolutional Network (GCN). The GCN follows the standard sparse message-passing formulation $H^{(\ell+1)} = \sigma(\hat{A}H^{(\ell)}W^{(\ell)})$, where $\hat{A}$ is a sparse normalized adjacency matrix. At each layer ℓ, the linear transformation $H^{(\ell)} \in \mathbb{R}^{N\times d}$ multiplied by $W^{(\ell)} \in \mathbb{R}^{d\times h}$ incurs a cost of $O(Ndh)$, while sparse aggregation with $\hat{A}$ costs $O(Eh)$, where $E \ll N^2$ is the number of edges. Thus, the total time complexity of an L_{gcn}-layer GCN is

$$T_{\text{GCN}} = O(L_{\text{gcn}}\,(Ndh + Eh))\,. \tag{20}$$

Transformer-based LLM Head. The LLM classifier is a Transformer encoder with batch size B, sequence length S, hidden dimension d_{model}, and L_{tf} layers. Each layer performs self-attention with complexity $O(BS^2 d_{\text{model}})$ and a feed-forward network with cost $O(BSd_{\text{model}}^2)$. Since $S \gg d_{\text{model}}$ in our setting, the attention term dominates, yielding

$$T_{\text{LLM}} = O\big(L_{\text{tf}}\,B\,S^2\,d_{\text{model}}\big) \tag{21}$$

Algorithm 1 Joint GCN–LLM Training with Early Stopping

Require: Node features X, adjacency matrix A; tokenized inputs (I, M); multi-hot labels Y; index sets $\mathcal{I}_{\text{train}}, \mathcal{I}_{\text{val}}$; learning rate η; patience p; maximum epochs E.
Ensure: Optimal parameters $\theta^{\star}_{\text{gcn}}, \theta^{\star}_{\text{llm}}$.

Initialize parameters $\theta_{\text{gcn}}, \theta_{\text{llm}}$
Initialize best validation loss $\mathcal{L}_{\text{best}} \leftarrow \infty$
Initialize patience counter $c \leftarrow 0$
for $e = 1, \ldots, E$ **do**
  Zero gradients: $\nabla_{\theta_{\text{gcn}}} \leftarrow 0,\ \nabla_{\theta_{\text{llm}}} \leftarrow 0$
  GCN forward (full graph):

$$Z_{\text{gcn}} = \text{GCN}_{\theta_{\text{gcn}}}(A, X)$$

  Select training logits:

$$Z^{\text{train}}_{\text{gcn}} = Z_{\text{gcn}}[\mathcal{I}_{\text{train}}]$$

  LLM forward (training subset only):

$$Z^{\text{train}}_{\text{llm}} = \text{LLM}_{\theta_{\text{llm}}}(I_{\text{train}}, M_{\text{train}})$$

  Compute joint training loss:

$$\mathcal{L}_{\text{train}} = \text{BCEWithLogits}\Big(Z^{\text{train}}_{\text{gcn}}, Y_{\text{train}}\Big) + \text{BCEWithLogits}\Big(Z^{\text{train}}_{\text{llm}}, Y_{\text{train}}\Big)$$

  Backward pass and parameter update:

$$(\theta_{\text{gcn}}, \theta_{\text{llm}}) \leftarrow (\theta_{\text{gcn}}, \theta_{\text{llm}}) - \eta \nabla \mathcal{L}_{\text{train}}$$

  Validation forward:

$$Z^{\text{val}}_{\text{gcn}} = \text{GCN}_{\theta_{\text{gcn}}}(X, A)[\mathcal{I}_{\text{val}}]$$

$$Z^{\text{val}}_{\text{llm}} = \text{LLM}_{\theta_{\text{llm}}}(I_{\text{val}}, M_{\text{val}})$$

  Compute validation loss:

$$\mathcal{L}_{\text{val}} = \text{BCEWithLogits}\Big(Z^{\text{val}}_{\text{gcn}}, Y_{\text{val}}\Big) + \text{BCEWithLogits}\Big(Z^{\text{val}}_{\text{llm}}, Y_{\text{val}}\Big)$$

  if $\mathcal{L}_{\text{val}} < \mathcal{L}_{\text{best}}$ **then**
    $\mathcal{L}_{\text{best}} \leftarrow \mathcal{L}_{\text{val}}$
    $c \leftarrow 0$
    Save model parameters $(\theta^{\star}_{\text{gcn}}, \theta^{\star}_{\text{llm}})$
  else
    $c \leftarrow c + 1$
    if $c \geq p$ **then**
      break
    end if
  end if
end for=0

Combined Training Cost per Epoch. Each training epoch consists of one forward and one backward pass through both the GCN and the LLM, with gradients computed from the summed loss. Therefore, the total time complexity per epoch is

$$T_{\text{epoch}} = O\Big(2\, L_{\text{gcn}}\,(Ndh + Eh) + 2\, L_{\text{tf}}\, B\, S^2\, d_{\text{model}}\Big) \tag{22}$$

B.2.1 Memory Complexity

GCN Memory. The GCN stores node embeddings of size $O(Nh)$, a sparse adjacency matrix of size $O(E)$, and a small number of layer parameters. Thus, its memory complexity is

$$S_{\text{GCN}} = O(Nh + E) \tag{23}$$

LLM Memory. The Transformer stores token embeddings and intermediate activations for backpropagation across all layers, resulting in a memory cost of

$$S_{\text{LLM}} = O(BSd_{\text{model}}L_{\text{tf}}) \tag{24}$$

Peak Memory Usage. Since the GCN and LLM are trained jointly, peak memory usage must accommodate both components:

$$S_{\text{peak}} = S_{\text{GCN}} + S_{\text{LLM}} = O(Nh + E + BSd_{\text{model}}L_{\text{tf}}) \tag{25}$$

Implementation Notes. In our implementation, the LLM processes the entire training set in a single batch, i.e., $B = |\mathcal{I}_{\text{train}}|$. When GPU memory is limited, this batch can be split into smaller sub-batches without affecting the asymptotic complexity. The GCN leverages sparse–dense multiplication ($E \ll N^2$), and operations such as dropout, bias addition, and nonlinear activations incur only $O(Nh)$ overhead per layer.

B.2.2 Complexity Analysis

Figure 4 compares the inference latency and memory footprint of our four models across the arXiv, DBLP, Elsevier, and PubMed datasets. RPCG-2 shows the highest runtime ($\approx 8s$ on arXiv) and peak memory usage ($\approx 23GB$), reflecting the cost of its enriched graph structure, while RPCG-1 is slightly more efficient due to fewer edge types. The lightweight GCN provides sub-second inference and minimal memory ($\approx 8GB$) but sacrifices classification capacity. The LLM-based model sits in the middle, moderate speed ($\approx 2s$) and memory ($\approx 12GB$), highlighting the fundamental trade-off between model complexity and deployability in citation-graph classification.

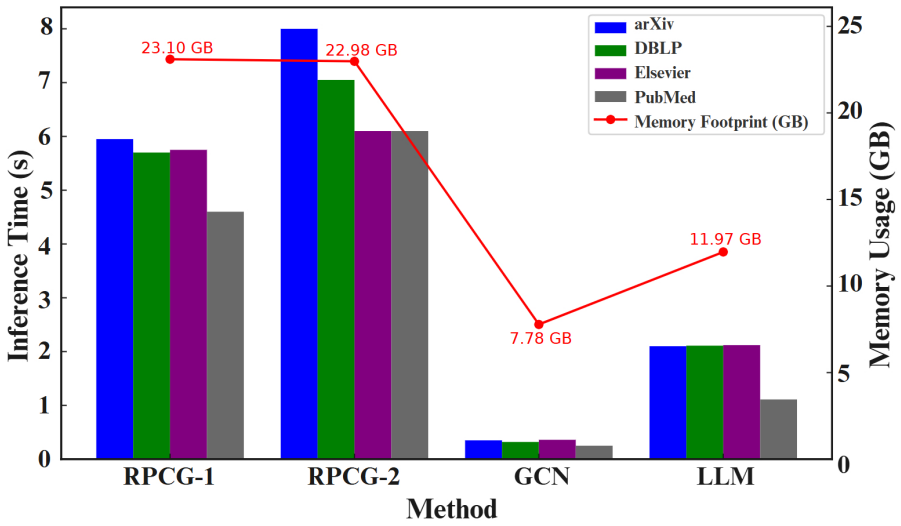


Figure 4: Usage memory and inference time of different models.

B.3 Mathematical Details and Proofs

B.3.1 Paper–Word–Citation Graph: Formulation and Proofs

Nodes. Let $\mathcal{P} = \{p_1, \ldots, p_{N_p}\}$ be the paper set and $\mathcal{W} = \{w_1, \ldots, w_V\}$ be the vocabulary. We construct a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \mathcal{P} \cup \mathcal{W}$ and $N = |\mathcal{V}| = N_p + V$.

Edges. The edge set $\mathcal{E}$ is the union of three relations used in the code:

(i) Citation edges. For a pair of papers (p_u, p_v), we add a directed citation edge $e = (p_u, p_v) \in \mathcal{E}_{pp}$ if paper p_u cites p_v (as stored in the JSON graph with `(citing → cited)`).

(ii) Paper–word TF–IDF edges. Let $T \in \mathbb{R}^{N_p \times V}$ be the TF–IDF document–term matrix. For paper p_j and term w_i, we add an edge $e = (p_j, w_i) \in \mathcal{E}_{pw}$ with weight

$$\omega_{pw}(p_j, w_i) = \text{tf}(p_j, w_i) \cdot \log\Big(\frac{N_p}{\text{df}(w_i)}\Big) \tag{26}$$

which matches the implementation `weight_val = tf * log(N/df)`

(iii) Word–word co-occurrence (PPMI) edges. Using a sliding window of size s over tokenized documents, let $C(i,j)$ be the co-occurrence count of (w_i, w_j) across all windows, $C(i)$ be the occurrence count of w_i, and W be the total number of windows. We define PMI as in the code:

$$\mathrm{PMI}(i,j) = \log\Big(\frac{C(i,j)\,W}{C(i)\,C(j)}\Big), \qquad \omega_{ww}(w_i,w_j) = \max\{\mathrm{PMI}(i,j), 0\} \tag{27}$$

and add undirected edges (w_i, w_j) and (w_j, w_i) whenever $\omega_{ww}(w_i, w_j) > 0$.

B.4 Paper–Author–Citation Graph: Formulation and Proofs

Nodes. Let $\mathcal{P} = \{p_1, \ldots, p_{N_p}\}$ be the set of papers and $\mathcal{A} = \{a_1, \ldots, a_{N_a}\}$ be the set of unique authors extracted from metadata. We define the paper–author heterogeneous graph

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}), \qquad \mathcal{V} = \mathcal{P} \cup \mathcal{A}, \qquad |\mathcal{V}| = N = N_p + N_a \tag{28}$$

Edges. The edge set is the union of citation edges and paper–author membership edges:

$$\mathcal{E} = \mathcal{E}_{pp} \cup \mathcal{E}_{pa}, \qquad \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \tag{29}$$

Citation edges. A directed edge $(p_u, p_v) \in \mathcal{E}_{pp}$ indicates that paper p_u cites paper p_v.

Paper–author edges. If author a_j appears in the author list of paper p_i, then $(p_i, a_j) \in \mathcal{E}_{pa}$. (If the implementation uses an undirected/symmetrized adjacency, we also include (a_j, p_i).)

Weighted adjacency. Let $A \in \mathbb{R}^{N \times N}$ be the sparse weighted adjacency matrix induced by $\mathcal{E}_{pp} \cup \mathcal{E}_{pw} \cup \mathcal{E}_{ww}$, where $A_{uv} = \omega(u,v)$ if $(u,v) \in \mathcal{E}$ and $A_{uv} = 0$ otherwise. We add self-loops and normalize exactly as in the code:

$$\hat{A} = A + I, \qquad D_{ii} = \sum_{j=1}^{N} \hat{A}_{ij}, \qquad \tilde{A} = D^{-\frac{1}{2}} \hat{A} D^{-\frac{1}{2}} \tag{30}$$

B.5 Data Labeling

We derive multi-label supervision from TF–IDF. Let $T \in \mathbb{R}^{N_p \times V}$ be TF–IDF. We compute corpus-level term scores and select Top-K terms:

$$s_i = \sum_{j=1}^{N_p} T_{j,i}, \qquad \{\pi(1), \ldots, \pi(K)\} = \arg\operatorname*{top\text{-}K}_i \; s_i \tag{31}$$

Each paper p_j is assigned a binary label vector $Y_j \in \{0,1\}^K$ by

$$Y_{jk} = \mathbb{I}\big(w_{\pi(k)} \in p_j\big), \qquad Y \in \{0,1\}^{N_p \times K} \tag{32}$$

B.5.1 GCN Forward Pass and Structural Logits

Let $X \in \mathbb{R}^{N \times d}$ be the initial node feature matrix (constructed from textual representations in the implementation). An L-layer GCN computes:

$$H^{(0)} = X, \qquad H^{(\ell)} = \sigma\Big(\tilde{A} H^{(\ell-1)} W^{(\ell)} + b^{(\ell)}\Big) \quad (1 \le \ell \le L-1) \tag{33}$$

and outputs node-wise logits (no activation in the last layer):

$$Z_{\mathrm{gcn}} = H^{(L-1)} W^{(L)} \in \mathbb{R}^{N \times K} \tag{34}$$

We train on the paper indices $\mathcal{I}_{\mathrm{train}} \subseteq \{1, \ldots, N_p\}$ by selecting $Z_{\mathrm{gcn}}^{\mathrm{train}} = Z_{\mathrm{gcn}}[\mathcal{I}_{\mathrm{train}}]$.

B.5.2 LLM Forward Pass and Semantic Logits

For papers in $\mathcal{I}_{\text{train}}$, we form token ids $I_{\text{train}} \in \mathbb{Z}^{|\mathcal{I}_{\text{train}}|\times S}$ and masks $M_{\text{train}} \in \{0,1\}^{|\mathcal{I}_{\text{train}}|\times S}$. The pretrained transformer classifier produces logits:

$$Z_{\text{llm}}^{\text{train}} = \text{LLM}_{\theta_{\text{llm}}}(I_{\text{train}}, M_{\text{train}}) \in \mathbb{R}^{|\mathcal{I}_{\text{train}}|\times K} \tag{35}$$

B.5.3 Joint Training Objective (BCEWithLogitsLoss)

The code optimizes both branches with `BCEWithLogitsLoss`, i.e., binary cross-entropy applied to logits. For logits $Z \in \mathbb{R}^{|\mathcal{I}_{\text{train}}|\times K}$ and labels $Y \in \{0,1\}^{|\mathcal{I}_{\text{train}}|\times K}$, the per-entry stable form is:

$$\mathcal{L}_{\text{BCELogits}}(Z,Y) = \frac{1}{|\mathcal{I}_{\text{train}}|K} \sum_{i=1}^{|\mathcal{I}_{\text{train}}|} \sum_{k=1}^{K} \Big[\max(Z_{ik},0) - Z_{ik}Y_{ik} + \log\big(1+\exp(-|Z_{ik}|)\big)\Big] \tag{36}$$

This loss is equivalent to standard BCE on $\sigma(Z_{ik})$ (sigmoid), but is numerically stable.

Branch losses and joint loss. Let $Y_{\text{train}} = Y[\mathcal{I}_{\text{train}}]$ be the Top-K label matrix. The structural and semantic losses are:

$$\mathcal{L}_{\text{gcn}} = \mathcal{L}_{\text{BCELogits}}(Z_{\text{gcn}}^{\text{train}}, Y_{\text{train}}), \qquad \mathcal{L}_{\text{llm}} = \mathcal{L}_{\text{BCELogits}}(Z_{\text{llm}}^{\text{train}}, Y_{\text{train}}) \tag{37}$$

and the code uses the unweighted sum:

$$\mathcal{L} = \mathcal{L}_{\text{gcn}} + \mathcal{L}_{\text{llm}} \tag{38}$$

B.5.4 Proofs (Code-Consistent)

Proposition 1 (PPMI edges are added iff co-occurrence exceeds independence). For any word pair (i,j), an edge is added iff $\text{PMI}(i,j) > 0$. **Proof.** From (27), $\text{PMI}(i,j) > 0$ holds iff $\frac{C(i,j)}{W} > \frac{C(i)}{W}\frac{C(j)}{W}$, i.e., the empirical co-occurrence probability exceeds the independence baseline. The implementation checks `if pmi > 0` and then adds both directions.

Proposition 2 (Paper–author edges are cross-type only). For any edge $(u,v) \in \mathcal{E}_{pa}$, one endpoint is a paper and the other is an author:

$$(u,v) \in \mathcal{E}_{pa} \;\Rightarrow\; (u \in \mathcal{P},\, v \in \mathcal{A}) \text{ or } (u \in \mathcal{A},\, v \in \mathcal{P}) \tag{39}$$

Proof. By construction, $\mathcal{E}_{pa}$ is created only from the membership relation "author a is listed in paper p". Hence endpoints must be of different node types.

Proposition 3 (Two-hop paths induce co-authorship). Define the co-authorship relation $a_u \sim a_v$ iff there exists a paper p authored by both. Then

$$a_u \sim a_v \quad\Longleftrightarrow\quad \exists\, p \in \mathcal{P} :\; (p,a_u) \in \mathcal{E}_{pa} \;\wedge\; (p,a_v) \in \mathcal{E}_{pa} \tag{40}$$

which corresponds to the length-2 path $a_u \leftarrow p \rightarrow a_v$ in $\mathcal{G}$. **Proof.** ($\Rightarrow$) If a_u and a_v co-author p, then both membership edges exist. ($\Leftarrow$) If both membership edges exist for some p, then a_u and a_v appear in p's author list, so they co-author p.

Proposition 4 (Paper–paper coupling via shared authors). If two papers p_i and p_j share at least one author, then there exists a two-hop walk $p_i \rightarrow a \rightarrow p_j$. Consequently, the (i,j) entry of $\hat{A}^2$ is positive:

$$\exists\, a \in \mathcal{A} :\; (p_i,a) \in \mathcal{E}_{pa},\; (p_j,a) \in \mathcal{E}_{pa} \;\Longrightarrow\; (\hat{A}^2)_{p_ip_j} > 0 \tag{41}$$

Proof. Matrix multiplication gives $(\hat{A}^2)_{p_ip_j} = \sum_v \hat{A}_{p_iv}\hat{A}_{vp_j}$. Choosing $v = a$ yields $\hat{A}_{p_ia}\hat{A}_{ap_j} > 0$ whenever both edges exist, hence the sum is positive.

Proposition 5 (Joint training yields gradient additivity across branches). Let θ_{gcn} and θ_{llm} be disjoint parameter sets. Then

$$\nabla_{\theta_{\text{gcn}}}\mathcal{L} = \nabla_{\theta_{\text{gcn}}}\mathcal{L}_{\text{gcn}}, \qquad \nabla_{\theta_{\text{llm}}}\mathcal{L} = \nabla_{\theta_{\text{llm}}}\mathcal{L}_{\text{llm}} \tag{42}$$

Proof. By (38), $\mathcal{L} = \mathcal{L}_{\text{gcn}} + \mathcal{L}_{\text{llm}}$. Since $\mathcal{L}_{\text{llm}}$ does not depend on θ_{gcn} and $\mathcal{L}_{\text{gcn}}$ does not depend on θ_{llm}, the cross-partials are zero and (42) follows. This matches the code that computes `loss = loss_gcn + loss_llm` and calls `loss.backward()` once.

Proposition 6 (Logit averaging corresponds to averaging log-odds). At inference, the implementation combines logits by $Z_{\text{fuse}} = \frac{1}{2}(Z^{\text{test}}_{\text{gcn}} + Z^{\text{test}}_{\text{llm}})$. Then for each label k, the fused logit equals the arithmetic mean of the two log-odds. **Proof.** For any logit z, the log-odds of sigmoid probability is $\log \frac{\sigma(z)}{1-\sigma(z)} = z$. Hence averaging logits is equivalent to averaging log-odds from the two branches (no additional fusion parameters are introduced in the code).

C Experimental Settings

In this section, we provide more details on the experimental environment, hyperparameter configurations, and dataset distribution.

C.0.1 Experimental environment

All experiments are conducted on a high-performance computing system equipped with an Intel Core i9-13900K processor (13th generation, 24 cores, 32 threads, base clock 3.00 GHz), NVIDIA GeForce RTX 4090 GPU with 24 GB of dedicated VRAM, and 128 GB of DDR5 RAM. The system also included a 4.55 TB SSD, all within a Windows 11(pro) environment. Key libraries such as PyTorch, PyTorch Geometric, Transformers, NumPy, SciPy, Pandas, Scikit-learn, and NLTK are used to support the development, graph construction, embedding extraction, and training of the proposed CiteGCN-LLM framework.

C.0.2 Hyperparameter Configurations

For GCN, we use an embedding dimension of 200 in the initial convolutional layer and a sliding window size of 20. Initial experiments showed robustness to slight changes, so we set the learning rate at 0.01, a dropout rate of 0.5, and an ℓ_2 regularization weight of 0. We randomly reserve 10% of the training set for validation. For LLM, we used AdamW with a learning rate of 2e-5, an epsilon of 1e-8, and a weight decay of 0.2. The scheduler has a warm-up of 0 steps. We train for up to 50 epochs using the Adam optimizer, including early stopping if the validation loss hasn't improved for 15 consecutive epochs.

C.0.3 Dataset Distribution

Figure 5 shows key statistics from our four citation datasets (arXiv, DBLP, Elsevier, and PubMed). Each row displays, on the left, the distribution of abstract lengths, which mostly range from 100 to 300 tokens, indicating fairly consistent document sizes. The middle panel presents citation-count histograms, showing a heavy-tailed distribution: while most papers have few citations, a small number receive many. In the right panel, author-degree distributions peak at 1–3 authors per paper, with some reaching 8 and more in collaborative works. Together, these visuals highlight the variety in text length, citation influence, and collaboration patterns that our models need to handle.

D Performance Evaluations

In this section, we provide more in-depth experimental results, discuss model limitations, and provide a discussion.

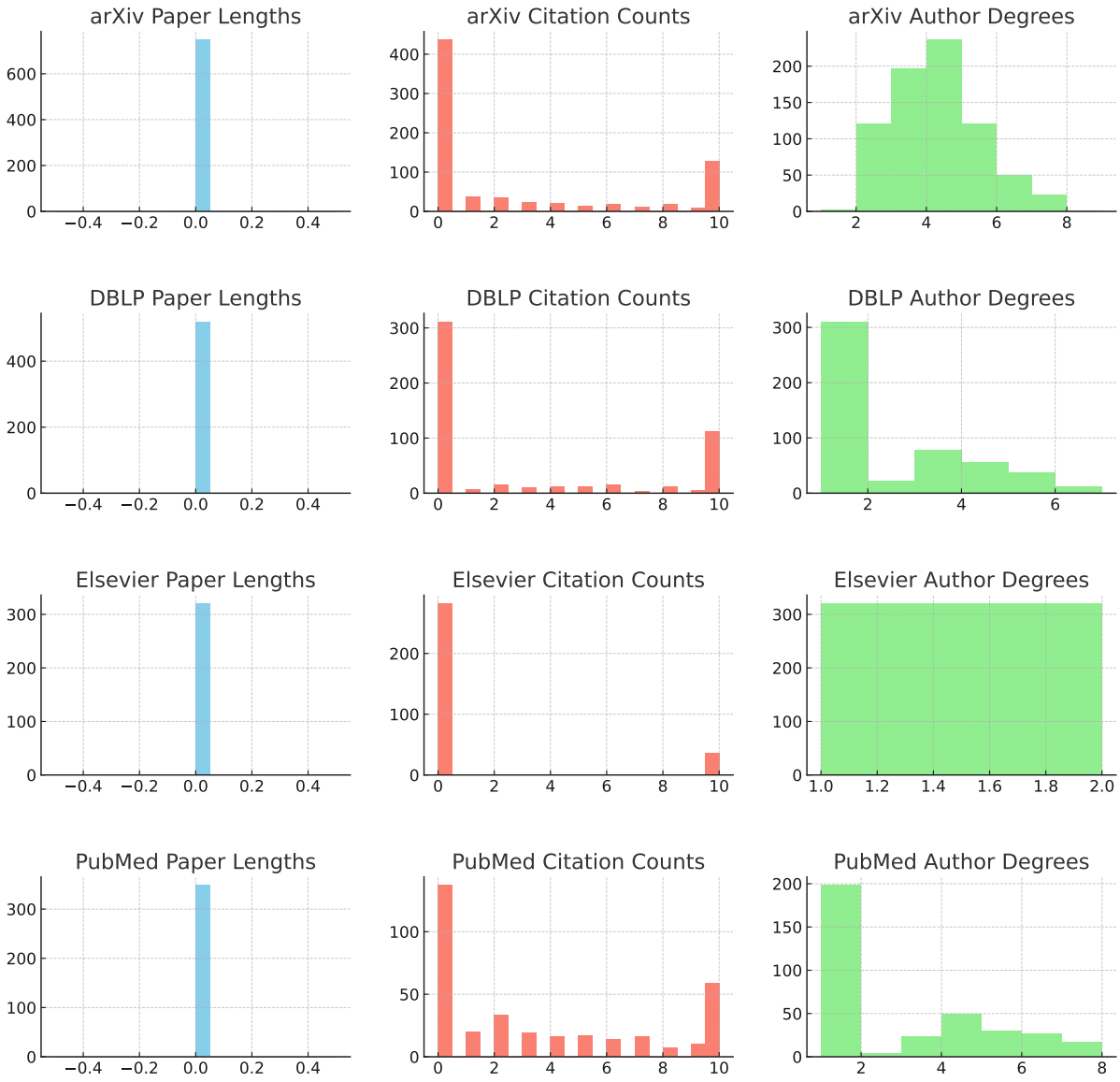


Figure 5: Distributions of paper (abstract) lengths, citation counts, and author degrees across arXiv, DBLP, Elsevier, and PubMed datasets.

D.0.1 Embedding Dimension Tuning

Figure 6 shows how GCN embedding dimensions affect classification accuracy for both (a) RPCG-1 and (b) RPCG-2 across four datasets. In both cases, accuracy rises quickly with low-dimensional representations, reaching a peak at 150–200 dimensions, exceeding 93% on arXiv, nearly 92% on Elsevier, and about 91% on DBLP before slowly decreasing beyond 200 dimensions. PubMed consistently performs lower, peaking at around 84%, likely due to its broader topic coverage. These graphs suggest that a mid-range embedding size offers the best balance between representation power and overfitting, as the additional gains beyond 200 dimensions diminish.

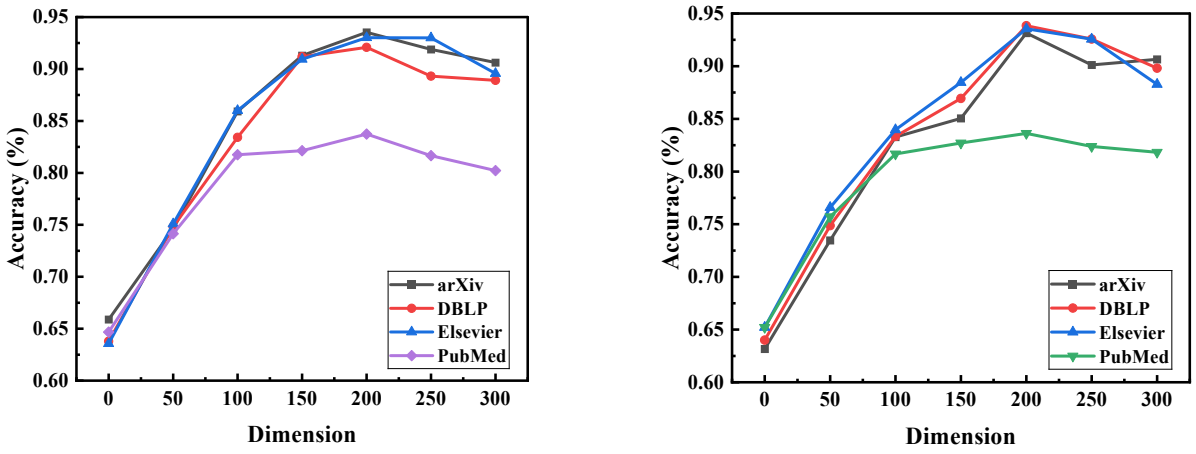


Figure 6: Impact of GCN embedding dimension size on the classification performance of RPCG-1 and RPCG-2.

D.0.2 Ablation Study

Table 4 shows the best Accuracy and F1 scores (mean ± std) for both RPCG-1 and RPCG-2, comparing a simple GCN backbone with our LLM-enhanced version across four citation datasets. In every setup, using transformer-pooled text embeddings consistently improves results. For example, RPCG-1 on arXiv increases from 90.20 ± 0.50% to 91.80 ± 0.42% in Accuracy and from 89.75 ± 0.45% to 91.40 ± 0.40% in F1 score. Similar improvements of 0.6 to 1.3 points in both metrics are seen on the DBLP, Elsevier, and PubMed datasets, with the largest gain on PubMed, where textual differences are greatest. Among the graph models, RPCG-2+LLM slightly outperforms RPCG-1+LLM on DBLP and Elsevier, highlighting the benefit of author-centric edges in collections rich in academic network data. These results show that training GCN propagation together with LLM embeddings leads to better classification results across different scholarly graphs.

Model	Backbone	arXiv		DBLP		Elsevier		PubMed	
		Acc.↑	F1↑	Acc.↑	F1↑	Acc.↑	F1↑	Acc.↑	F1↑
RPCG-1	GCN	90.20 (0.50)	89.75 (0.45)	88.80 (0.48)	88.30 (0.44)	86.50 (0.30)	86.10 (0.32)	75.00 (0.40)	74.60 (0.38)
	LLM	91.80 (0.42)	91.40 (0.40)	90.80 (0.38)	90.50 (0.36)	88.60 (0.29)	88.30 (0.31)	80.00 (0.36)	79.70 (0.34)
RPCG-2	GCN	90.50 (0.48)	90.10 (0.43)	89.00 (0.45)	88.60 (0.41)	87.00 (0.28)	86.70 (0.30)	75.50 (0.35)	75.10 (0.33)
	LLM	91.30 (0.45)	91.00 (0.42)	89.50 (0.40)	89.20 (0.38)	87.20 (0.32)	86.90 (0.34)	78.80 (0.38)	78.50 (0.36)

Table 4: Comparison of GCN and LLM backbones' performance for RPCG-1 and RPCG-2 across our citation datasets

D.0.3 Limitations and Discussion

Despite achieving high-performing results, CiteGCN-LLM has several practical limitations. First, both RPCG-1 and RPCG-2 rely on static citation and co-authorship networks, which fail to

capture developing research trends or shifts in author influence. Adding dynamic graph updates could help the model stay current as new publications and collaborations appear. Second, our LLM embeddings depend solely on titles and abstracts, overlooking valuable intra-document signals such as full-text content, figures, tables, and citation contexts. Extending the encoder to include multimodal or full-text inputs could improve semantic understanding, though it would increase computational costs. Third, the Top-K labeling strategy we use may underrepresent niche or emerging topics that fall outside the most common labels. This underscores the need for adaptive, user-driven label selection methods like reinforcement learning or Bayesian personalization to reflect evolving researcher interests better. Fourth, while LLM and GCN reduce per-query costs, the entire pipeline remains resource-intensive for large datasets, and storing multiple heterogeneous graphs requires significant memory. Techniques such as graph sampling, subgraph training, or lightweight transformer architectures could improve scalability. Lastly, our evaluation is limited to English-language benchmarks. To ensure the robustness and fairness of CiteGCN-LLM across diverse scholarly communities, it will be essential to test it using low-resource or multilingual datasets with cross-lingual embedding alignment.