Refining High-Quality Labels Using Large Language Models to Enhance Node Classification in Graph Echo State Network

Ikhlas Bargougui[1,3], Rebh Soltani[1] and Hela Ltifi[1,2]
[1]*Research Groups in Intelligent Machines, University of Sfax, National Engineering School of Sfax (ENIS), BP 1173, 3038, Sfax, Tunisia*
[2]*Computer Science and Mathematics Department, Faculty of Science and Technology of Sidi Bouzid, University of Kairouan, Sidi Bouzid, 9100, Tunisia*
[3]*Computer Science and Mathematics Department, Faculty of Economics and Management Sciences, University of Sfax, Sfax, 3018, Tunisia*

Keywords: Large Language Models, Graph Echo State Network, Reservoir Computing, Node Classification.

Abstract: Graph learning has attracted significant attention due to its applicability in various real-world scenarios involving textual data. Recent advancements, such as Graph Echo State Networks (GESN) within the reservoir computing (RC) paradigm, have shown notable success in node-level classification tasks, especially for heterophilic graphs. However, graph neural networks (GNNs) suffer from the need for a large number of high-quality labels to achieve promising performance. Conversely, large language models (LLMs), with their extensive knowledge bases, have demonstrated impressive zero-shot and few-shot learning abilities, particularly for node classification tasks. However, LLMs struggle with efficiently processing structural data and incur high inference costs. In this paper, we introduce a novel pipeline named LLM-GESN, which involves four flexible components: k-means clustering for active node selection, LLM for difficulty aware annotation, adaptable post-selection, and GESN model training and prediction. Experimental results demonstrate the effectiveness of LLM-GESN on text-attributed graphs from the Cora, CiteSeer, Pubmed, Wikics, and ogbn-arxiv datasets. Our LLM-GESN achieved significant test accuracy of 86.67%, 76.63%, 74.58%, 77.09%, and 58.79%, respectively, compared to state-of-the-art methods.

1 INTRODUCTION

Deep learning (DL) has revolutionized numerous machine learning tasks, including image classification (Li S. S., 2019), video processing (Sharma, 2021), speech recognition (Soltani, Newman-Watts-Strogatz topology in deep echo state networks for speech emotion recognition, 2024), and natural language processing (Otter, 2020). The majority of DL models involve data in Euclidean space. However, many applications, such as paper citations, web page links, social network interactions, and molecular bonds, involve data from non-Euclidean domains represented as graphs. The complexity and richness of graph-structured data, along with the availability of large datasets, have driven a surge in developing DL models for adaptive graph processing. In recent years, GNNs have been widely employed in various graph analysis tasks, including node-focused tasks (e.g., node classification, link prediction) and graph-focused tasks (e.g., graph similarity detection, graph classification) (Abadal, 2021).

Node classification is an essential area of research due to its wide range of applications, one of the best-known traditional pipelines for this task are: the first is the GNN-based pipeline that involves a fixed training set with truth labels and the GNN is trained on these truth labels from the graph then predicts the labels of the rest of the unlabeled nodes in the test phase (Huang, 2020), the second graph active learning-based pipeline aims to select a group of nodes from the pool in order to maximize the performance of the GNN models trained on these graphs with labels (Chen Z. M., 2023). Both pipelines neglect the subtleties of the annotation process which can be both costly and error-prone in practice, even for simple tasks. Moreover, this can be even more difficult if active learning of graphs is taken into account; with improvements in annotation diversity

Bargougui, I., Soltani, R. and Ltifi, H.
Refining High-Quality Labels Using Large Language Models to Enhance Node Classification in Graph Echo State Network.
DOI: 10.5220/0013220600003890
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 2*, pages 555-565
ISBN: 978-989-758-737-5; ISSN: 2184-433X
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

inevitably increasing the difficulty of ensuring annotation quality for these reasons other pipelines appears. In order to overcome these limitations, authors of (Chen Z. M., 2023) propose a pipeline capable of exploiting LLM to automatically generate high-quality annotations and use them to train a GNN model with promising performance. LLMs, in contrast to GNN, excel in zero-shot and few-shot node classification on TAGs with unlabeled nodes. Nonetheless, they struggle to capture graph structural patterns as effectively as GNNs. LLMs are also limited by input context length, reducing their ability to utilize extensive labels for fine-tuning. Consequently, LLMs may underperform compared to well-trained GNNs and have higher prediction costs, making them less scalable for large datasets. However, most GNNs utilize a layered architecture that performs local aggregation of node features. This local aggregation progressively smooth node features in deeper layers, creating difficulties in GNN models (Xie Y. L., 2020). This bias towards locally homogeneous graphs is particularly problematic in node classification tasks involving graphs with a large number of inter-class edges, or a low degree of homophily. The GESN was introduced by (Micheli, 2023). In GESN, input data is encoded via a randomly initialized reservoir, requiring learning only at the linear readout stage, which enhances efficiency and mitigates the issues arising from local feature aggregation.

In this paper, we exploit the strengths of LLM and GESN in node classification by investigating the potential of exploiting the zero-shot learning capabilities of LLM to alleviate the substantial training data demands of GESN while addressing their inherent weaknesses. Our LLM-GESN involves four flexible components: k-means clustering for active node selection, LLM for difficulty aware annotation, adaptable post-selection, and GESN model training and prediction. LLMs are utilized to annotate a small subset of nodes, which are then used to train GESNs to predict the remaining nodes. This approach presents unique challenges: selecting the most beneficial nodes for LLM annotation to optimize GESN training, and ensuring that LLM annotations are of high quality, representativeness, and diversity to enhance GESN performance.

The remaining of this paper is organized as follows: Section 1 provides an overview of related works, focusing on Graph Echo State Networks (GESN) for graph node classification and the use of Large Language Models (LLMs) in graph structures. Section 2 introduces our proposed method in detail. Section 3 presents the evaluation of our model on node classification tasks ranging from medium- to large-scale graphs with different degrees of heterophily., including experimental settings and a comprehensive performance. Finally, we conclude our study and discuss potential future directions in Section 4.

2 RELATED WORKS

2.1 Graph Echo State Network for Graph Nodes Classification

Echo State Networks (ESNs) (Soltani, 2023) (Soltani, 2024) (Soltani, 2024)are recurrent randomized neural networks that feature a large, fixed, recurrent reservoir. In this framework, there are three layers: input layer, reservoir layer and the readout layer. Only the weights of the readout layer are trained.

When applied to graph structures, GESN computes graph representations, or embedding, as fixed points of a dynamical system. Initially introduced as the GESN (Micheli, 2023) (Jellali, 2023), this approach was later extended into deep architectures (Micheli, 2023). GESN was applied in the literature, on all types of graph tasks including graph classification, edge classification and node classification. Recent research has explored graph classification to predict graph molecular toxicity using GESN. Gallicchio et al. (Gallicchio, 2020) introduced a constructive algorithm for GESN, applying reservoir computing to graph classification domains for toxicology tasks.

Moreover, GESN has emerged as an efficient approach for processing dynamic graphs. The Dynamic Graph Echo State Network (DynGESN) model has been proposed for discrete-time dynamic graphs, offering competitive accuracy and improved efficiency compared to temporal graph kernels and convolutional networks (Tortorella, 2021).This approach provides vector encodings for dynamic graphs without requiring training, making it suitable for large-scale data. Building on this concept, the Grouped Dynamical Graph Echo State Network (GDGESN) introduces a snapshot merging strategy to enhance performance in spreading process classification tasks (Li Z. F., 2023). These reservoir computing-based models offer a balance between predictive performance and computational efficiency.

GESN has also shown promise in node classification tasks on graphs. Traditional approaches involve fixed connections in the hidden layer and training only the output weights (Gallicchio, 2020).

However, recent research suggests that output weight training can be omitted in favor of supervised clustering based on principal components of reservoir states, potentially improving classification accuracy (Prater, 2016). GESN have demonstrated effectiveness in addressing heterophily challenges in node classification, outperforming many fully trained deep models (Micheli, 2023). GESN computes node embeddings recursively using an untrained message-passing function, effectively encoding structural relationships.

2.2 Large Language Models for Graphs

LLMs are employed as **Enhancers** in graph learning by initially processing graph data enriched with textual information, thereby enhancing node embedding or labels to augment GNN training. This approach addresses issues with diverse initial node embedding, which may lack clarity and diversity, potentially leading to suboptimal GNN performance. Recent advancements leverage LLMs' language modelling capabilities to generate more meaningful and effective embedding for improved GNN training output. Several methods use this approach. For instance, G-Prompt added a GNN layer to pre-trained LLMs for task-specific node embedding using prompt tuning (Fang, 2024). SimTeG fine-tunes text embedding obtained from LLMs for node classification tasks using GNNs (Duan, 2023). GALM utilizes BERT for text embedding and employs unsupervised learning tasks to optimize model parameters for various applications (Xie H. Z., 2023). LLMRec enhances user-item interaction data using GPT-3.5 to enrich node embedding with rich textual profiles, thereby improving the performance of recommender systems (Wei, 2024).

LLMs also serve as **Labelers** in graph learning, where labels generated by LLMs are used to supervise GNN training. This method extends beyond simple categorical labels to include embedding and other forms of information. Leveraging LLMs in this manner provides supervision signals that aid in optimizing GNN performance across various graph-related tasks. Examples of this approach include OpenGraph, which uses LLMs to generate nodes and edges, addressing sparse data issues through sampling and refinement strategies (Xia, 2024). LLM-GNN employs LLMs to annotate nodes with confidence-scored category predictions, enhancing GNN training with diverse labels (Micheli, 2023). GraphEdit utilizes LLMs to predict and refine candidate edges in comparison to original graph edges, improving edge prediction accuracy (Guo, 2024).

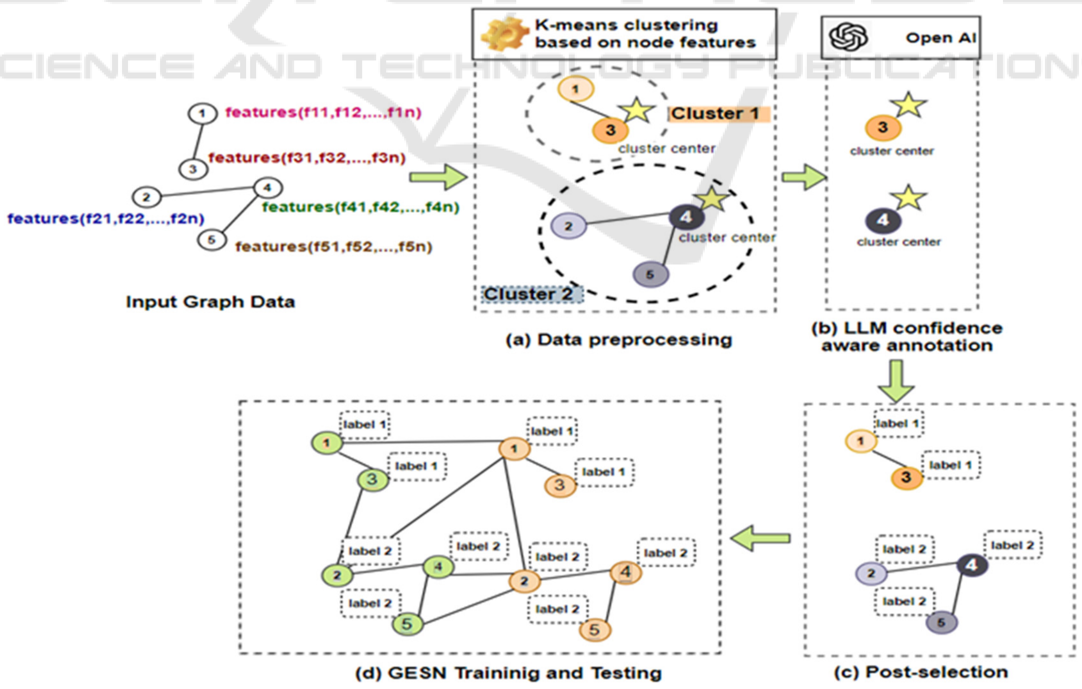
K-means clustering based on node features
Open AI
features(f11,f12,...,f1n)
features(f31,f32,...,f3n)
features(f21,f22,...,f2n)
features(f41,f42,...,f4n)
features(f51,f52,...,f5n)
Input Graph Data
Cluster 1
cluster center
cluster center
Cluster 2
(a) Data preprocessing
cluster center
cluster center
(b) LLM confidence aware annotation
label 1
label 2
(d) GESN Traininig and Testing
(c) Post-selection

Figure 1: Our proposed pipeline method LLM-GESN.

Despite the advancements made by these methods in enhancing graph learning performance, a notable limitation remains: the decoupling of LLMs from GNNs necessitates a two-stage learning process. This separation is often due to resource constraints posed by large graphs or complex LLM parameters, which strongly influences GNN performance reliant on pre-generated LLM embedding and task-specific prompts.

3 PROPOSED METHOD

Figure1 provides a detailed explanation of our proposed pipeline method. This pipeline consists of four main steps: data preprocessing, LLM confidence-aware annotation, post-selection, and GESN training.

3.1 Data Preprocessing

GESNs need numerous high-quality labels for optimal node classification performance. The main challenge is actively selecting nodes for annotation by LLMs to enhance GESN training. Leveraging LLMs for high-quality, representative, and diverse annotations can significantly improve GESN performance.

The active node selection process identifies a small set of candidate nodes for LLM annotation, maintaining a manageable budget and considering diversity and representativeness as the original baseline. It is also important to consider label quality, as LLMs can produce noisy labels with significant variance across different node groups. Therefore, establishing heuristics that link to the annotation difficulty of various nodes is crucial. A preliminary analysis of LLM annotations provides insights into how to infer annotation difficulty based on node features showing that the accuracy of annotations generated by LLMs is closely related to the clustering density of the nodes.

To demonstrate this correlation, k-means clustering, as shown in Figure 2, is applied to the original feature space, setting the number of clusters to match the distinct class count. We sample 1000 nodes from the entire dataset and annotate those using LLMs. These nodes are then sorted and divided into ten equal-sized groups based on their distance to the nearest cluster center. This distance serves as a heuristic to estimate annotation reliability.Given that the number of clusters matches the number of distinct classes, we refer to this heuristic as $\phi_{density}(V_i)$, calculated as:

$$\phi_{density}(V_i) = \frac{1}{1 + ED(EM_{V_i}, CC_{V_i})} \quad (1)$$

Where ED is the Euclidean distance metric, EM_{V_i} is the embedding of node v_i and CC_{V_i} is the center of the cluster that V_i belongs to. A higher value of $\phi_{density}(V_i)$ indicates that V_i more representative within the embedding space.

We then integrate this annotation difficulty heuristic into the active selection process, B Nodes in the unlabeled pool are chosen based on their highest scores. To enhance model performance, the selected nodes should balance annotation difficulty with traditional active selection criteria, such as representativeness and diversity (Zhang, 2021). These criteria can be expressed as a score function.

An effective method for integrating the difficulty heuristic into this learning process is ranking aggregation. This method is more robust to differences in scale because it only considers the relative order of the elements. We incorporate the difficulty heuristic by first converting $\phi_{density}(V_i)$ into a rank $r_{\phi_{density}}(V_i)$. Then, we combine these two rankings to obtain:

$$f_{DA-act}(V_i) = \alpha_0 \times r_{f_{act}}(V_i) + \alpha_1 \times r_{\phi_{density}}(v_i) \quad (2)$$

Where $f_{act}(v_i)$ is the initial score function for active graph learning, $r_{f_{act}}(v_i)$ is its ranking by decreasing percentage and "DA" means difficulty aware. The hyperprameters α0 and α1 allow us to balance sannotation difficulty with traditional criteria for active graph learning, such as representativeness and diversity. Nodes *vi* with higher $f_{DA-act}(V_i)$ scores are then selected for annotation by the LLMs, forming the V_{an} set.

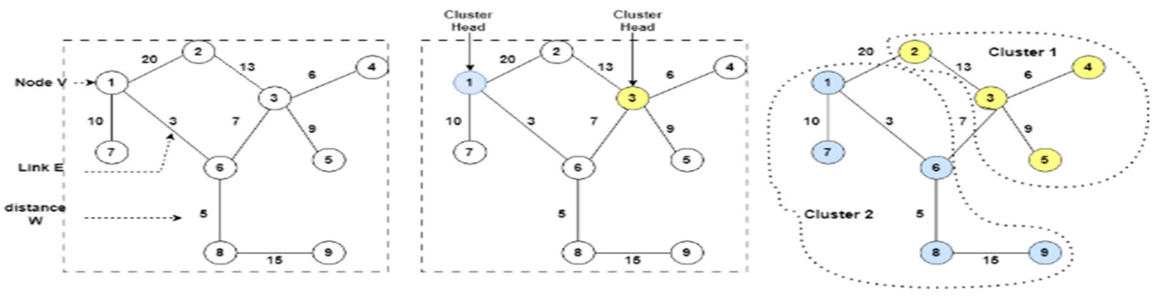

Figure 2: K-means clustering for active node selection.

3.2 LLM Confidence Aware Annotation

After selecting nodes (candidate set V_{an}) easily to annotate with difficulty-aware active node selection, we then utilize the strong zero-shot ability of LLMs to annotate those nodes with confidence-aware prompts. But LLM annotations, akin to human annotations, can exhibit a certain degree of label noise. That's means we are not aware of how the LLM responds to the nodes at that stage. It leads to the potential of remaining low-quality nodes. To figure out the high-quality annotations, we need some guidance on their reliability, such as the confidence scores that can help to identify the annotation quality and help us filter high-quality labels from noisy ones.

Inspired by recent literature on generating calibrated confidence from LLMs, we aim to identify the most effective prompts in terms of accuracy, confidence calibration, and cost-efficiency.

We observe that LLMs exhibit promising zero-shot prediction performance across all datasets. Zero-shot annotation is a technique where a model is used to annotate data without having seen examples of the specific annotation task during its training. The consistency strategy involves ensuring that the annotations are consistent across similar instances in the dataset. In Table 1 a comprehensive prompt example for zero-shot annotation with a consistency strategy.

Table 1: Complete prompt for zero-shot annotation with consistency strategy used across all datasets.

Input: Question: (Contents) Paper:
Title: #the title of the suggested paper#
Abstract: #the abstract of the suggested paper#
Task: There are following categories: [*category*^{1}, *category*^{2}, ..., *category*]#
What's the category of this paper?
Provide your 3 best guesses and a confidence number that each is correct (0 to 100) for the following question from most probable to least. The sum of all confidence should be 100.
For example, [{" answer": <your first answer>," confidence": <confidence for first answer>}, ...]
Output:

3.3 Post-Selection

The post-selection step aims to eliminate low-quality annotations by using confidence scores generated by the LLMs. By leveraging these scores, we refine the set of annotated nodes by removing those with lower confidence, ensuring the labels remain high-quality. However, directly filtering out low-confidence nodes can result in a shift in label distribution and reduce the diversity of the selected nodes, which may degrade the performance of the trained models.

During the post-selection stage, label distribution is readily available, allowing us to directly consider the label diversity of the selected nodes. To measure the change in diversity, a simple score function called Change of Entropy (COE) is proposed. This function measures the entropy change of labels when removing a node from the selected set and can be computed as follows:

$$COE(v_i)=\mathbf{H}(A_{V_{set}}-\{v_i\})-\mathbf{H}(A_{V_{set}}) \quad (3)$$

Where V_{set} is the current selected set of nodes, A present the annotation generated by LLMs and **H** () is the Shannon entropy function (Shannon, 1948).

The value of COE can be either positive or negative, with a small COE (v_i) value indicating that removing the node could negatively impact the diversity of the selected set, potentially affecting the performance of the trained models. When a node is removed, the entropy adjusts, requiring a re-computation of COE. This re-computation is computationally efficient since the size of the selected set V_{an} is much smaller than the entire dataset. COE can be combined with the confidence score $f_{conf}(v_i)$ to balance diversity and annotation quality through ranking aggregation, also $r_{\phi_{density}}(v_i)$ is available in the post-selection phase. The final selection score function can be calculated as:

$$f_{select}(v_i) = \delta_0 \times r_{f_{conf}(v_i)} + \delta_1 \times r_{COE(v_i)} + \delta_2 \times r_{\phi_{density}}(v_i) \quad (4)$$

Where $r_{f_{conf}}$ is the high-to-low ranking percentage of the confidence score f_{conf} . To conduct post-selection, each time we remove the node with the smallest f_{select} value until a specified maximum number is attained. $\delta_0, \delta_1, \delta_2$ are hyper-parameters introduced to balance label diversity and annotation quality.

3.4 GESN Training

GESN operates within the RC framework, where input data is processed by a randomly initialized reservoir, and only the linear readout layer requires training. This approach benefits from the dynamic properties of reservoirs, making it suitable for handling diverse graph structures.

Another crucial aspect of the training process is the choice of the loss function. While GNNs usually utilize cross-entropy loss, the presence of noisy labels from the LLMs makes a weighted cross-entropy loss more appropriate. By using the confidence scores from the previous section as weights, we can enhance the model's robustness and improve the overall quality of the annotations.

GESNs create whole graph embedding from node embedding using simple pooling functions like sum or mean. These node embedding are generated by updating a non-linear dynamical system iteratively, similar to GNN models.

According to Figure 3, Node embedding is recursively computed by the non-linear dynamical system:

$$\boldsymbol{h}_v^{(k)} = tanh\left(\boldsymbol{W}_{in}\boldsymbol{x}_v + \sum_{u \in N_1(v)} \widehat{\boldsymbol{W}}\, \boldsymbol{h}_u^{(k-1)}\right), \quad \boldsymbol{h}_v^{(0)} = \boldsymbol{0} \quad (5)$$

Where $W_{in} \in \mathbb{R}^{H\times X}$ and $\widehat{W} \in \mathbb{R}^{H\times H}$ are, respectively, the input-to-reservoir and the recurrent weights, for a reservoir with H units (input bias is omitted). Reservoir weights are initially randomized using a uniform distribution within the range $[-1,1]$. They are then rescaled to achieve the desired input scaling and reservoir spectral radius, all without the need for any training. Equation (1) is iterated over k up to K times, after which the final state $X_v^{(K)}$ is used as the node embedding.

For node classification tasks, a linear readout is applied to node embeddings:

$$\mathbf{y}_v = \mathbf{W}_{out}\mathbf{h}_v^{(k)} + \mathbf{b}_{out} \quad (6)$$

Where the weights $w_{out} \in \mathbb{R}^{H\times X}, \mathbf{b}_{out} \in \mathbb{R}^C$ are trained by ridge regression on one-hot encodings of target classesy_v.

The dynamical system (1) has been designed to be asymptotically stable, which means that it converges to a fixed point $h_v^{(\infty)}$ as $K \to \infty$. This convergence is ensured by the graph embedding stability (GES) property (Gallicchio, 2020), which also guarantees that the system is independent of its initial state $\boldsymbol{h}_v^{(0)}$. A sufficient condition for the GES property is to ensure that the transition function defined in (1) is contractive, i.e. has a Lipschitz constant such that $||\widehat{\mathbf{W}}||\ ||\mathbf{A}|| < \mathbf{1}$.

In standard reservoir computing, recurrent weights are initialized based on a necessary condition for the Generalized Echo State (GES) property, which is $\rho(w) < {}^{1}/_{\alpha}$, where ρ (-) denotes the spectral radius of a matrix, and $\alpha = \rho(A)$ is the spectral radius of the graph.

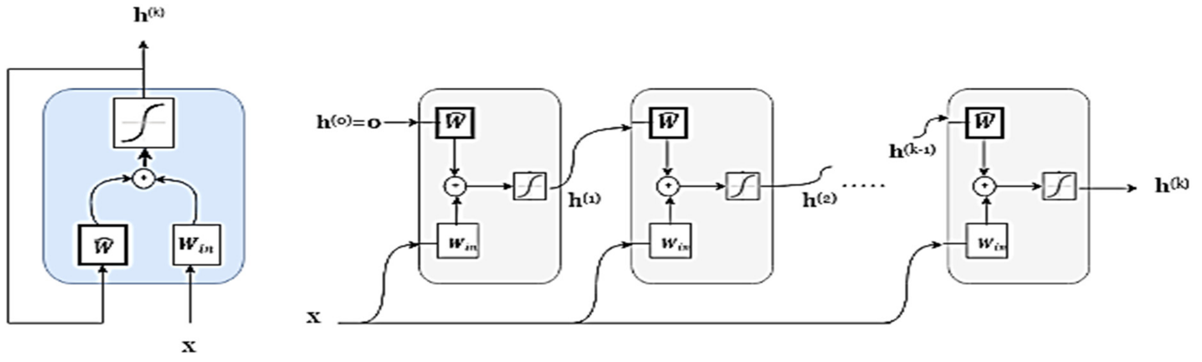

Figure 3: GESN general design.

Table 2: Dataset Characteristics and Metadata.

Datasets	Nodes	Edges	Tasks	Categories	Homophily	References
CORA	2,708	5,429	Categorize papers by title and abstract	7	0.81	(Chen Z. M., 2024)
CITESEER	3,186	4,277	Categorize papers by title and abstract	6	0.74	(Chen Z. M., 2024)
PUBMED	19,717	44,335	Categorize papers by title and abstract	3	0.80	(Chen Z. M., 2024)
WIKICS	11,701	215,863	Categorize articles by Wikipedia content	10	-	(Chen Z. M., 2024)
OGBN-ARXIV	169,343	1,166,243	Categorize papers by title and abstract	40	0.22	(Chen Z. M., 2024)

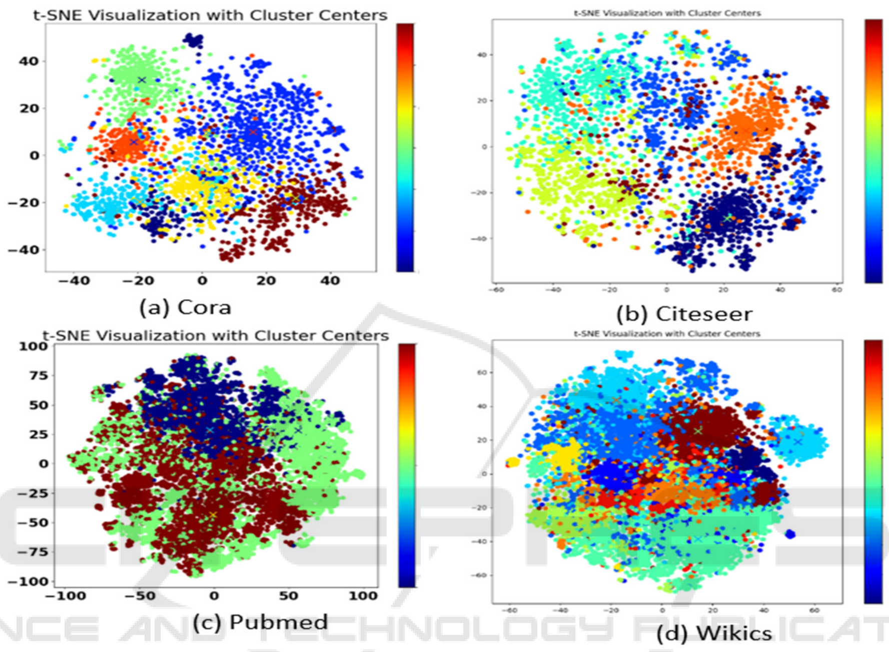

Figure 4: Clustered t-SNE Visualization: Grouping of Data Points Around Cluster Centers.

This condition provides the best estimate of the system's tipping point, beyond which equation (2) becomes asymptotically unstable.

4 EVALUATION

4.1 Experimental Settings

The experiment begins with an LLM-based annotation process for preprocessed graph nodes. This process involves formulating queries based on node features and local graph structure, which are then input to the LLM (gpt 3.5 turbo) to produce high quality labels. Following this, the GESN is employed for node classification. The GESN consists of a 2-layer with 256 hidden features. It utilizes sbert embedding as input and is initialized using carefully scaled uniform distributions. Our model incorporates a PageRank-based active learning strategy with varying budget constraints and employs both consistency-based and confidence-entropy filtering mechanisms.

4.2 Datasets

To assess the effectiveness of our model, we use five benchmark datasets: Cora, Citeseer, PubMed, WikiCS, and OGBN-Arxiv. All datasets are within the same domain of node classification. Table 2 details the main characteristics of these datasets.

4.3 Performance Analysis

4.3.1 Performance on Node Clustering

The t-SNE visualization with cluster centers, presented in Figure 4 shows how data points are grouped into clusters.

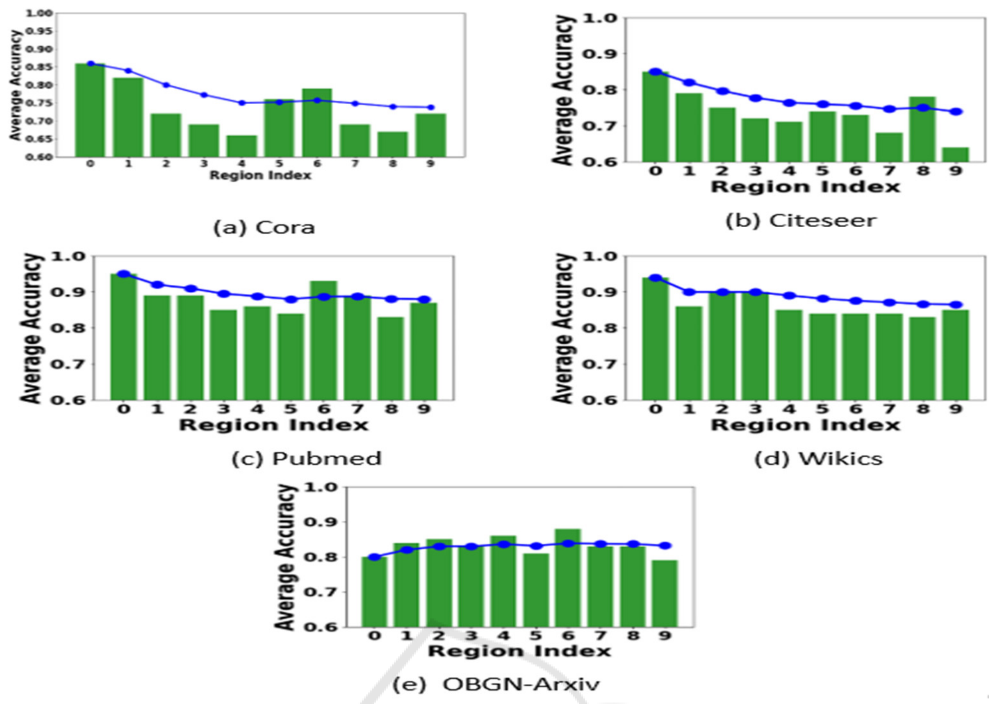

Figure 5: Relationship between LLM Annotation Accuracy and Node Distance from Cluster Center.

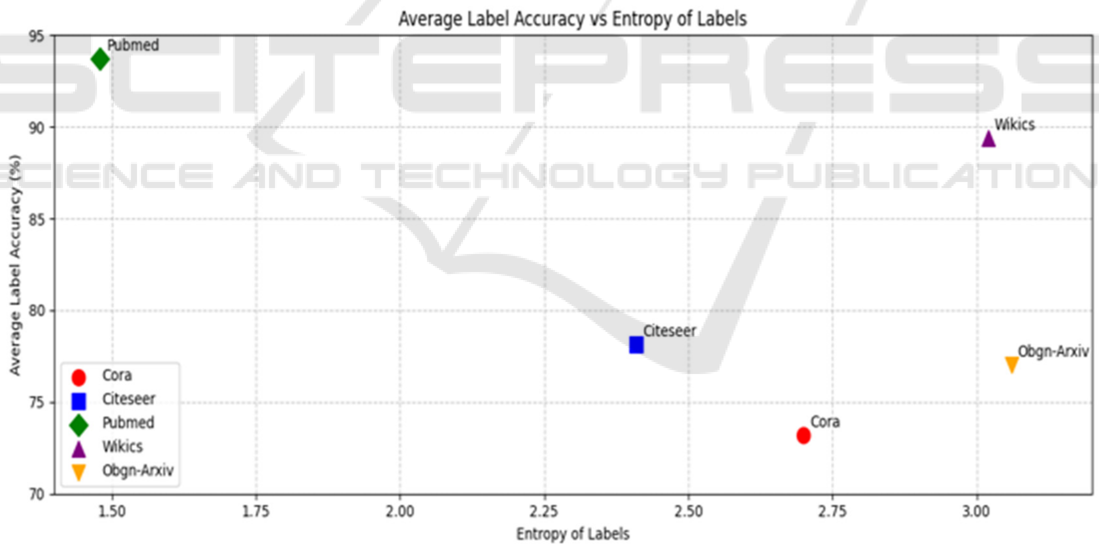

Figure 6: Correlation between Label Entropy and Average Label Accuracy across Datasets.

Each point is color-coded according to its assigned cluster (0 to # of classes - 1). The black crosses represent the center of these clusters.

By analyzing the distribution of points around each center, we can understand the cohesion and distinctiveness of the clusters. Here, the resulting figure for K-means clustering on the Arxiv dataset is omitted from our manuscript due to the complexity of visualizing 40 clusters, which would not provide clear interpretation.

4.3.2 Performance on Node Annotation

Based on Figure 5, in our proposed method, the accuracy of LLM annotation depends on the distance of nodes from the cluster center. When nodes are close to the center (lower region index), LLM annotation tends to be more accurate.

These central nodes likely share common features and context, making them easier for LLMs to annotate effectively. Conversely, distant nodes (higher region index) exhibit lower accuracy in LLM annotation.

To optimize LLM annotation quality, prioritizing nodes near the centroid becomes crucial. By doing so, we enhance the chances of accurate labeling.

In Figure 6, we explore the relationship between label entropy and average label accuracy across various datasets. The x-axis represents the entropy of labels within different datasets, quantifying the uncertainty or disorder in a set of labels, with higher entropy values indicating more diverse or ambiguous labels.

The y-axis represents the average label accuracy (in percentage) for each dataset, reflecting how accurately labels can be applied to data points Observations from the plot reveal that as the entropy of labels increases (moving right along the x-axis); average label accuracy tends to decrease. Datasets with lower entropy (more certain labels) exhibit higher accuracy, while those with higher entropy (more uncertain or diverse labels) experience lower accuracy.

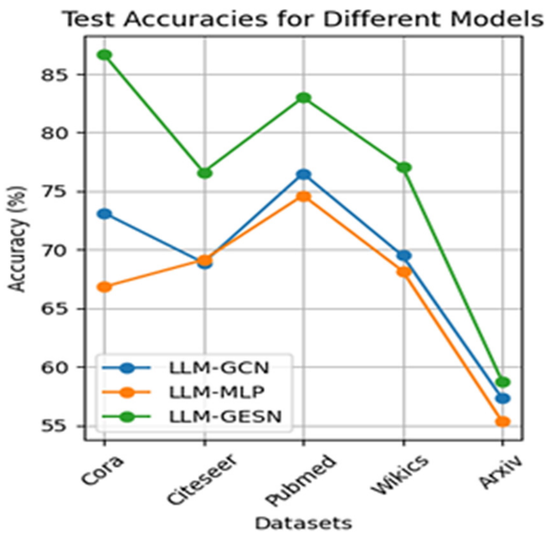

Figure 7: test accuracies for five different datasets across three models LLM-GCN, LLM-MLP and LLM-GESN.

4.3.3 Performance on Node Classification

The test accuracy in Figure 7 compares the performance of three models—LLM-GCN, LLM-MLP, and LLM-GESN—across five datasets: Cora, Citeseer, Pubmed, Wikics, and Arxiv. LLM-GESN

Table 3: Comparative Analysis: Our Model Versus State-of-the-Art Test Accuracies Across Datasets.

Reference	Model	Test accuracy
	Cora	
(Hou, 2022)	Self-supervised GraphMAE	84.2±0.4
(Micheli, 2023)	GESN	86.04±1.01
	LLM-GESN	**86.67 ± 0.00**
	Citeseer	
(Hou, 2022)	Self-supervised GraphMAE	73.4±0.4
(Micheli, 2023)	GESN	74.51±2.14
	LLM-GESN	**76.63 ± 0.00**
	PubMed	
(Hou, 2022)	GAE	72.1±0.5
(Arnaiz-Rodríguez, 2022)	GCN	68.19±0.7
	LLM-GESN	**74.58 ± 0.00**
	Wikics	
(Hoang, 2023)	GIN	76.53±0.82
	LLM-GESN	**77.09 ± 0.00**
	OBGN-Arxiv	
(Sharma, 2021)	GESN	48.80±0.22
	LLM-GESN	**58.79 ± 0.00**

competes well on Cora and Pubmed, but shows weaker results on Citeseer and Arxiv. These findings underscore the efficiency of LLM-GESN on all datasets.

As displayed in table 3, our proposed pipeline, LLM-GESN, is evaluated against state-of-the-art methods across all datasets (Cora, Citeseer, PubMed, WikiCS, and OBGN-Arxiv).

The results show that the LLM-GESN model outperforms other models in terms of test accuracy. For the Cora dataset, LLM-GESN achieved the highest accuracy of 86.67 ± 0.00, surpassing both Self-supervised GraphMAE and GESN. In the Citeseer dataset, LLM-GESN reached an accuracy of 76.63 ± 0.00, again outperforming Self-supervised GraphMAE and GESN. For the PubMed dataset, LLM-GESN recorded 74.58 ± 0.00, higher than both GAE and GCN. On the WikiCS dataset, LLM-GESN achieved 77.09 ± 0.00, slightly better than GIN. The most significant improvement was seen in the OBGN-Arxiv dataset, where LLM-GESN achieved 58.79 ± 0.00, far surpassing GESN. Overall, the LLM-GESN model demonstrates superior performance and robustness across all datasets, highlighting its effectiveness in enhancing test accuracies compared to state-of-the-art models. We note that the impact of LLM annotation to enhance the performance of GESN node classification on small to large-scale dataset as Cora, Citeseer and OBGN-Arxiv. Moreover, our model achieves respectable performance even comparing with other models like GraphMAE, GAE, GCN, GIN.

However, this model still lacks hyperparameter optimization for the GESN, which could further enhance these results. Tuning hyperparameters such as learning rates, regularization strengths, and network architectures could potentially improve the model's performance across different datasets. Additionally, in terms of real-world applications, deploying LLM-GESN could be transformative. For example, in social network analysis, LLM-GESN could accurately classify nodes based on their attributes, such as predicting user preferences or behaviors. This capability could aid in personalized recommendation systems or targeted marketing strategies, where understanding and predicting individual user characteristics are crucial for enhancing user engagement and satisfaction.

5 CONCLUSION

In this paper, we address two significant challenges in node classification for graph data as a prominent topic in data science: the issue of heterophilic graphs and the requirement of high-quality annotations. We propose a new model LLM-GESN that investigates the potential of harnessing the zero-shot learning capabilities of LLMs to alleviate the substantial training data demands of GESNs. Comprehensive experiments on graphs of various scales validate the effectiveness of our pipeline. Demonstrating that our model achieves accuracy comparable to or better than the GESN model and other GNN models that utilize LLMs for annotation. In future work, we plan to validate our model in real-world applications, recognizing the importance of hyperparameters optimization for GESN.

REFERENCES

Abadal, S. J.-A. (2021). Computing graph neural networks: A survey from algorithms to accelerators. *ACM Computing Surveys (CSUR), 54(9)*, 1-38.

Arnaiz-Rodríguez, A. B. (2022). Diffwire: Inductive graph rewiring via the lov\\'asz bound. *arXiv preprint arXiv:2206.07369.*

Chen, Z. M. (2023). Label-free node classification on graphs with large language models (llms). *arXiv preprint arXiv:2310.04668*.

Chen, Z. M. (2024). Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter, 25(2)*, 42-61.

Chen, Z. M. (2024). Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter, 25(2)*, 42-61.

Duan, K. L. (2023). Simteg: A frustratingly simple approach improves textual graph learning. *arXiv preprint arXiv:2308.02565*.

Fang, T. Z. (2024). Universal prompt tuning for graph neural networks. *Advances in Neural Information Processing Systems, 36*.

Gallicchio, C. &. (2020). Fast and deep graph neural networks. *In Proceedings of the AAAI conference on artificial intelligence (Vol. 34, No. 04, pp. 3898-3905).*

Guo, Z. X. (2024). Graphedit: Large language models for graph structure learning. *arXiv preprint arXiv:2402.15183*.

Hoang, V. T. (2023). Mitigating Degree Biases in Message Passing Mechanism by Utilizing Community Structures. *arXiv preprint arXiv:2312.16788.*

Hou, Z. L. (2022). Graphmae: Self-supervised masked graph autoencoders. *In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (pp. 594-604).

Huang, Q. H. (2020). Combining label propagation and simple models out-performs graph neural networks. *arXiv preprint arXiv:2010.13993.*

Jellali, N. S. (2023). An Improved Eulerian Echo State Network for Static Temporal Graphs. In International Conference on Intelligent Systems Design and

Applications. *Cham: Springer Nature Switzerland*, 307-318.

Li, S. S. (2019). *An overview: Deep learning for hyperspectral image classification.* IEEE Transactions on Geoscience and Remote Sensing, 57(9), 6690-6709. (2019).

Li, Z. F. (2023). Dynamical Graph Echo State Networks with Snapshot Merging for Spreading Process Classification. *In International Conference on Neural Information Processing (pp. 523-534). Singapore: Springer Nature Sin*.

Li, Z. F.-5. (n.d.).

Micheli, A. &. (2023). Addressing heterophily in node classification with graph echo state networks. *Neurocomputing, 550, 126506*.

Otter, D. W. (2020). A survey of the usages of deep learning for natural language processing. *IEEE transactions on neural networks and learning systems, 32(2),*, 604-624.

Prater, A. (2016). Reservoir computing for spatiotemporal signal classification without trained output weights. *arXiv preprint arXiv:1604.03073*.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal, 27(3)*, 379-423.

Sharma, V. G. (2021). *Video processing using deep learning techniques: A systematic literature review.* IEEE Access, 9, 139489-139507.

Soltani, R. B. (2023). Echo state network optimization: A systematic literature review. *Neural Processing Letters, 55(8)*, 10251-10285.

Soltani, R. B. (2024). Hybrid Quanvolutional Echo State Network for Time Series Prediction. *In ICAART (2)*, (pp. 40-46).

Soltani, R. B. (2024). *Newman-Watts-Strogatz topology in deep echo state networks for speech emotion recognition.* Engineering Applications of Artificial Intelligence, 133, 108293.

Soltani, R. B. (2024). Topology-adaptive Bayesian optimization for deep ring echo state networks in speech emotion recognition. *Neural Computing and Applications*, 1-18.

Tortorella, D. &. (2021). Dynamic graph echo state networks. *arXiv preprint arXiv:2110.08565*.

Wei, W. R. (2024). Llmrec: Large language models with graph augmentation for recommendation. *In Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, (pp. 806-815).

Xia, L. K. (2024). Opengraph: Towards open graph foundation models. *arXiv preprint arXiv:2403.01121.*

Xie, H. Z. (2023). Graph-aware language model pre-training on a large graph corpus can help multiple graph applications. *In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* , (pp. 5270-5281).

Xie, Y. L. (2020). When do gnns work: Understanding and improving neighborhood aggregation. *In IJCAI'20: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence,{IJCAI}*.

Zhang, W. Y. (2021). Grain: Improving data efficiency of graph neural networks via diversified influence maximization. *arXiv preprint arXiv:2108.00219*.