Interacted Graph-Language Models on Citation Networks Response to Reviewers' Comments

Anonymous Author(s)
Submission Id: 1321

A1. USAGE OF CHATGPT

For the prompt inputs of LLMs, we adhere to established methods [1, 6]. These prompt inputs consist of the paper's abstract and title, accompanied by a task-specific question. The objective of the question is to prompt the LLM to predict the paper's class and provide a corresponding explanation for the prediction. The textual responses generated by the LLMs will subsequently serve as inputs for the SLMs (in Sec. 5.2). In other words, the SLMs will use text responses generated by LLMs to generate text embeddings for all nodes. Then, the obtained text embeddings will be further processed by GNNs (in Sec. 5.1). For reference, we provide examples of the prompt inputs for each dataset below:

- Cora:
  Abstract: < `abstract text` >
  Title: < `title text` >
  Question: Which of the following sub-categories of AI does this paper belong to: Case Based, Genetic Algorithms, Neural Networks, Probabilistic Methods, Reinforcement Learning, Rule Learning, Theory? If multiple options apply, provide a comma-separated list ordered from most to least related, then for each choice you gave, explain how it is present in the text.
- Pubmed:
  Abstract: < `abstract text` >
  Title: < `title text` >
  Question: Does the paper involve any cases of Type 1 diabetes, Type 2 diabetes, or Experimentally induced diabetes? Please give one or more answers of either Type 1 diabetes, Type 2 diabetes, or Experimentally induced diabetes; if multiple options apply, provide a comma-separated list ordered from most to least related, then for each choice you gave, give a detailed explanation with quotes from the text explaining why it is related to the chosen option.
- arxiv23:
  Abstract: < `abstract text` >
  Title: < `title text` >
  Question: Which arXiv CS sub-category does this paper belong to? Give 5 likely arXiv CS sub-categories as a comma-separated list ordered from most to least likely, in the form "cs.XX", and provide your reasoning.
- ogbn-arxiv:
  Abstract: < `abstract text` >
  Title: < `title text` >
  Question: Which arXiv CS sub-category does this paper belong to? Give five likely arXiv CS sub-categories as a comma-separated list ordered from most to least likely, in the form "cs.XX", and provide your reasoning.

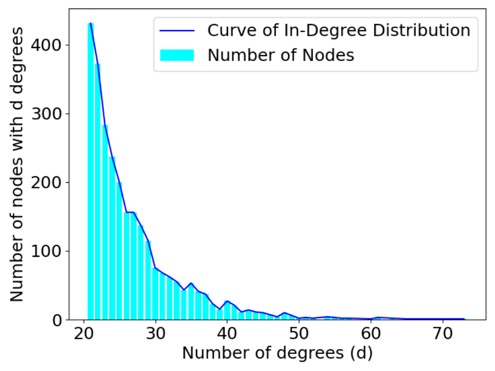


Figure 1: In-degree distribution of the graph generated by SLAPS on the Cora dataset.

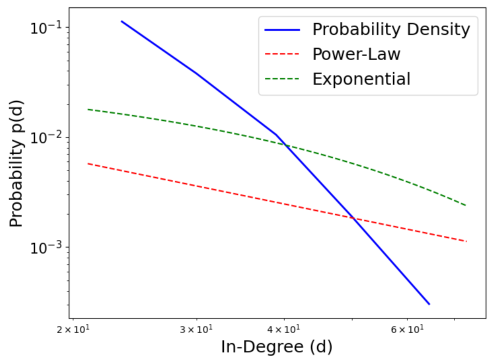


Figure 2: Fitting to the in-degree distribution of the SLAPS graph on the Cora dataset.

A2. METHOD COMPARISON

A2.1. Methods Mentioned by Reviewer GYUe

In fact, we compare with GLEM [7] in our experiments. GLEM applies DeBERTa and finetune it on labeled nodes. For GNNs, they select different GNNs, such as GCN and GraphSAGE. In our experiments, we chose the GCN version as all our baselines. As shown in Sec. 6.1.3, Line 658, we simply term GELM with GCN and DeBERTa as GCN+DeBERTa. This is convenient for comparing the specific language and GNN models used in our experiments. Besides, we adopt TAPE [1] for comparison because TAPE is an advanced version of GLEM with ChatGPT. To the best of our knowledge, TAPE [1] is currently the SOTA method on citation networks, which was accepted by ICLR 24.

Although GLEM [7] also uses pseudo labels to make the language models to generate better embeddings, the targets they focused on, the challenges they faced, and the outcomes they made are all totally different from ours. Specifically, GLEM focuses on how to integrate both the text and graph structure information with language models and GNNs. The challenges they faced is how to combine language models and GNNs into a joint framework. The outcome they made is an effective variational expectation maximization framework that updates language models and GNNs in the E-step and M-step.

However, we focus on how to find a good structure prior for both latent graph generation and improved finetuning of language models. The challenges we faced is that there are very limited labels for training, and the structure prior is required to be reasonable—it should be an inherent attribute of the data, and simple—the graph with such a structure prior should be easy to construct (see Sec 3.3). The outcomes we made are: 1) we identify such a structure prior as scale-free property and uncover that it can be modeled by a simple KNN graph, 2) we integrate such structure prior into a well-designed mutually reinforcing training strategy, and 3) we demonstrate the synergistic potential between LMs and GNNs even without the graph structure.

More importantly, GLEM requires both graph structure and node features to be available during training. When the graph structure is missing, GLME cannot be directly applied for citation networks.

A2.2. Methods Mentioned by Reviewer JUpD

The first paper introduces the BAGL [3] to infer the weights in the optimal graph structure. It is an optimization method that requires training on the data to obtain the optimal graph. But our method does not need additional training to obtain the latent graph. Moreover, in BAGL, the optimality of the affinity graph is built upon the label inference, but it may not be consistent with the true graph structure of the real-world data. Our method can obtain a meaningful graph structure that approximates the scale-free property of real-world citation networks.

The second paper proposes the VPL [8] to address the presence of degenerate solutions that tend to be almost constant for unlabeled points. It is an optimization-driven method that adopts Laplace learning and Poisson learning for graph-based semi-supervised learning. Our work focuses on exploring a good structure prior for both latent graph generation and improved finetuning of language models. Moreover, the VPL method requires the edge weight matrix to be available and the semantic information of node features is not fully utilized, while our method aims to infer a meaningful latent graph and extract better text embeddings for citation networks.

A2.3. Methods Mentioned by Reviewer FASr

The first paper introduces NodeFormer [5], a novel all-pair message passing scheme for efficiently propagating node signals between arbitrary nodes. The second paper proposes DIFFormer [4], an energy-driven geometric diffusion model with latent diffusivity function for data representations. Both NodeFormer and DIFFormer can learn/construct latent graphs efficiently. However, fast graph learning is not the core goal of our paper. We would like to emphasize that this paper focuses on how to find a good structure prior for both latent graph generation and improved finetuning of language models. The structure prior is required to be reasonable—it should be an inherent attribute of the data, and simple—the graph with such a structure prior should be easy to construct (see Sec 3.3). Although NodeFormer and DIFFormer achieve high efficiency on graph generation, it is hard to guarantee that the generated graphs look like scale-free networks.

The third paper introduces Tail-GNN [2], which learns a neighborhood translation from the structurally rich head nodes to enhance the representations of tail nodes. Tail-GNN requires the real graph structure of data to be available so that the tail and head nodes can be separated and processed with different operations. However, in our setting, the real graph structure of data is inaccessible. Besides, the goal of Tail-GNN is to utilize the graph structure and the node features of data to enhance the representations of tail nodes, while our goal is to infer a reasonable graph to enhance the finetuning of language models.

A2.4. Methods Mentioned by Reviewer AVgA

At first, we would like to clarify that this paper aims to jointly solve the two problems on modeling citation networks, i.e., 1) how to design an effective pseudo labeler to improve the finetuning of language models (as we empirically observed that existing LAG models suffer from limited labels, see Sec 3.1), and 2) how to efficiently infer the underlying structure information of the data (as existing graph structure learning models require specific optimization and additional training on the whole data, causing high model and time complexity, see Sec 3.2). The reason why we adopt the KNN graph is that it obeys two-fold characteristics: reasonable (it approximates the underlying structure of citation networks) and efficient (no additional training on the datasets).

It is worth noting that, besides KNN, we can adopt any graph structure learning (GSL) method in our training framework to generate pseudo labels. However, compared with KNN graphs, existing GSL methods require specific optimization strategies with complex constraints and need additional training on the whole data, causing high model and time complexity. In addition, it is hard to guarantee that the graphs generated from existing GSL methods are scale-free networks. Taking SLAPS as an example, it requires several hours of training to get the graphs while KNN only needs several minutes for graph construction. Moreover, as we can see from Figs 1 and 2, the graph generated by SLAPS on the Cora dataset does not approximate a scale-free property of citation networks.

A2.5. Methods Mentioned by Reviewer Vjfp

In fact, preferential attachment theory (PAT) assumes that there exists some edges in the networks so that the probabilities for the new edges can be calculated based on existing edges. This assumption is totally different from ours as there are no known edges in our problem setting. Moreover, PAT adds edges node by node until all nodes are connected. We cannot follow this way to generate the graph structure as it is extremely time-consuming for large citation networks. Instead, we adopt KNN to generate all edges at once, and we find that the degree of generated edges approximately follow the power-law distribution. Besides, PAT describes the edge generation process and explains why the degrees of real-world networks follow the power-law distribution. However, we don't care about why real-world networks look like that. We only care about whether the graphs generated by our model meet the structure prior we expect, that is, its degrees conform to the power-law distribution.

A3. PARAMETER AND TRAINING

For all the models employed in our experiments, we utilize consistent parameter settings to ensure fairness. For the GNNs, we employ a two-layer GCN with a hidden dimension of 128. The learning rate is set to 0.001, the dropout ratio is set to 0.5, and the weight decay is set to 0.0005. For the SLMs, we adopt a pretrained DeBERTa model and configure the dropout ratio as 0.3, the learning rate as 2×10^{-5}, the batch size as 20, and the weight decay as 0. We will make our codes and checkpoints publicly available.

In the first training stage, the hand-crafted node features and the KNN graphs will be input to the GNN for training. After training, we predict pseudo labels for unlabeled nodes and use it for training the LMs. Then the GNN will be trained again with the text embeddings generated by LMs and the KNN graphs, and generate the final predictions.

It is possible to train the GNNs and LM jointly. But at first, we need to train the GNN at least once before training the whole model because the finetuning of LMs requires pseudo labels in the case of limited supervision.

A4. MODEL ARCHITECTURE

Figure 6 shows the three main stage of the proposed method, which including pseudo labeling, textual modeling, and text embedding. In the pseudo labeling stage, we use nodes features and KNN graphs to train a GCN for pseudo labeling of unlabeled nodes. In the textual Modeling stage, we input text attributes of nodes into a LLM to get the text responses. In the text embedding stage, the text responses and pseudo labels generated from the first two stages will be input into a SLM to generate text embeddings for all nodes.

A5. RESULTS USING DIFFERENT K

In fact, the values of K have a significant impact on the distribution of degrees. In our method, the in-degree distribution of KNN graphs depends on the value of K, the similarity metric, and the text encoders used. In Figs. 4 and 5, we show the degree distribution when K=20. As we can see, the degree distributions are more like the power-law than exponential on the Cora and ogbn-arxiv datasets.

A6. RECONSTRUCTION OF CITATION NETWORKS

We would like to emphasize that our target is to explore the real structure prior of citation networks to improve the training of both LMs and GNNs for classification, rather than reconstruct all true edges in the dataset (All latent graph inference methods mentioned in Sec. 2.3 do not focus on reconstructing all true edges either). In KNN, whether to construct an edge between two nodes only depends on their feature similarity in a specific feature space. The feature similarity is measured by the similarity metric we used and the feature space is determined by the text encoders we adopted. Therefore, the accuracy of reconstructing known edges depends on the value of K, the similarity metric, and the text encoders. In fact, with the same similarity metric and text encoders, we can achieve higher accuracy to reconstruct more known edges by choosing a larger K (For example, we can reconstruct 2700 and 3422 edges on the Cora dataset when K=25 and K=50 respectively). It is also worth noting that, using all known edges does not guarantee the optimal prediction performance. As we can see from Table 4, using KNN graphs can achieve competitive even better performance than using true graphs.

A7. ABUNDANT LABELS

Please note that one of our goals is to improve the finetuning of language models under limited labels. If there are abundant labels, language models will be much stronger than GNNs. In this case, it is unnecessary to use GNNs for pseudo labeling. For illustration, we use 18217 labels of the Pubmed dataset for training. The results of GCN, DeBERTa, IntGL, DeBERTa+ChatGPT, and IntGL+ChatGPT are 0.8092 ± 0.0057, 0.9510 ± 0.0056, 0.9382 ± 0.007, 0.9388 ± 0.0020, and 0.9400 ± 0.0019. As we can see, only using DeBERTa can achieve the optimal performance. If we still use GCN to generate pseudo labels for DeBERTa, the performance will degrade. This is because GCN underperforms than DeBERTa in the case of abundant labels. As a result, the pseudo labels are of low quality, impairing the performance of LMs. On the other hand, the performance of GNNs can be improved as LMs can provide high-quality text embeddings for GNNs.

Algorithm 1 IntGL Algorithm

```
Require: X: feature matrix of nodes, Y_L: labels of labeled nodes,
    k the number of neighbors, q: general question, p_i: prompt.
 1: Stage 1: Graph-To-Language
 2:     Feature Encoding: $\mathbf{F} = \mathsf{Enc}(\mathbf{X})$.
 3:     KNN Graph Generation: $\mathbf{A} = \mathsf{KNN}_{\cos}(\mathbf{F}, k)$
 4:     GNN Training: $\mathsf{GNN}_{\Phi}(\mathbf{F}, \mathbf{A})$ using Eq. (3) with labels $\mathbf{Y}_L$.
 5:     Pseudo Labeling: $\hat{\mathbf{y}}_j = \mathrm{argmax}(\mathsf{GNN}_{\Phi}(\mathbf{f}_j, \mathbf{A})),\ j \in \{1, \cdots, n\}$.
 6:
 7: Stage 2: Large Language Models (optional)
 8:     Textual Response: $\mathbf{r}_i = \mathsf{LLM}(\mathbf{p}_i), \quad \mathbf{p}_i = \mathrm{concat}(\mathbf{x}_i, \mathbf{q})$
 9:
10: Stage 3: Language-To-Graph
11:     SLM Finetuning: $\mathsf{SLM}_{\Theta}(\mathbf{x}_i)$ using Eq. (5) with true labels $\mathbf{Y}_L$
12:                     and pseudo labels $\hat{\mathbf{Y}}_U = [\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_n]$.
13:     Text Embeddings: $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_{m+n}]$, where $\mathbf{e}_i = \mathsf{SLM}_{\Theta}(\mathbf{x}_i)$.
14:     (If LLM is used, we replace $\mathbf{x}_i$ with $\mathbf{r}_i$ in Lines 11 and 12.)
15:
16: Finetune GNN: $\mathsf{GNN}_{\Phi}(\mathbf{E}, \mathbf{A})$ using Eq. (3) with labels $\mathbf{Y}_L$.
17: Label Prediction: $\overline{\mathbf{y}}_j = \mathrm{argmax}(\mathsf{GNN}_{\Phi}(\mathbf{e}_j, \mathbf{A})),\ j \in \{1, \cdots, n\}$.
18:
19: return $\mathbf{Y}_U = [\overline{\mathbf{y}}_1, \cdots, \overline{\mathbf{y}}_n]$
```

REFERENCES

[1] Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. 2024. Harnessing Explanations: LLM-to-LM Interpreter for Enhanced Text-Attributed Graph Representation Learning. *International Conference on Learning Representations* (2024).
[2] Zemin Liu, Trung-Kien Nguyen, and Yuan Fang. 2021. Tail-GNN: Tail-Node Graph Neural Networks. In *Proceedings of the 27th ACM SIGKDD Conference*

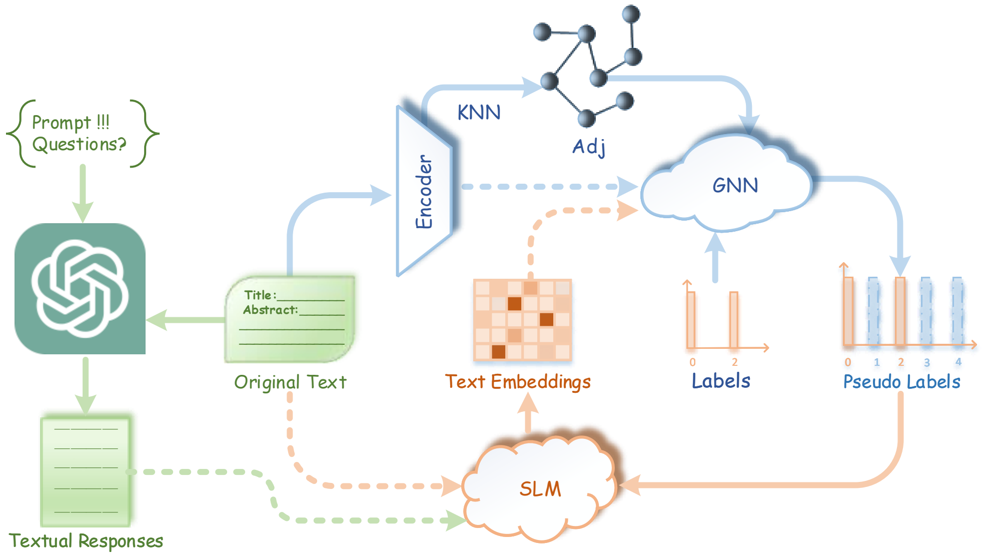
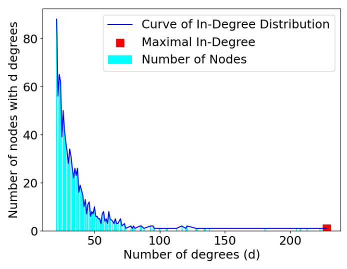
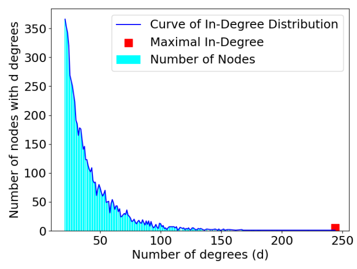


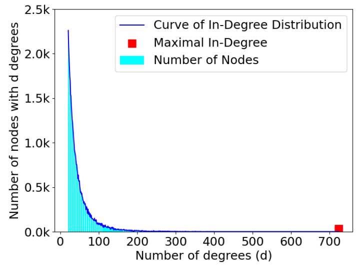
Figure 3: Framework of the proposed IntGL. We first generate a KNN graph and input it with node features into the GNN for training. After training, we generate pseudo labels for unlabeled nodes. Then, all pseudo and true labels will be used as supervision for finetuning the SLM on the original text or the textual responses generated from the LLM. After finetuning, we generate text embeddings using SLM and use them to retrain the GNN. We then adopt the well-trained GNN for final prediction of unlabeled nodes.



(a) Cora

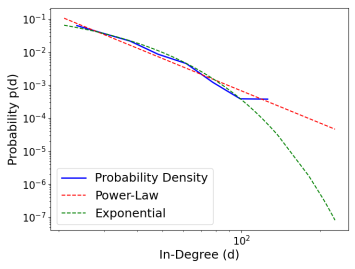


(b) Pubmed

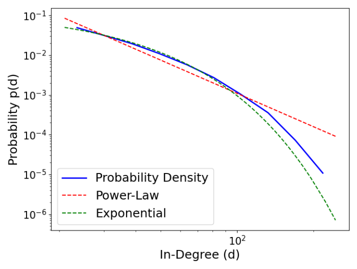


(c) ogbn-arxiv

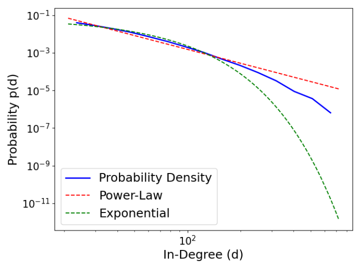
Figure 4: In-degree distribution of KNN graphs on different datasets, where the maximal hub is marked in red.



(a) Cora



(b) Pubmed



(c) ogbn-arxiv

Figure 5: Fitting to the in-degree distribution of different datasets with power-law and exponential distributions.

on Knowledge Discovery & Data Mining (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 1109–1119. https: //doi.org/10.1145/3447548.3467276

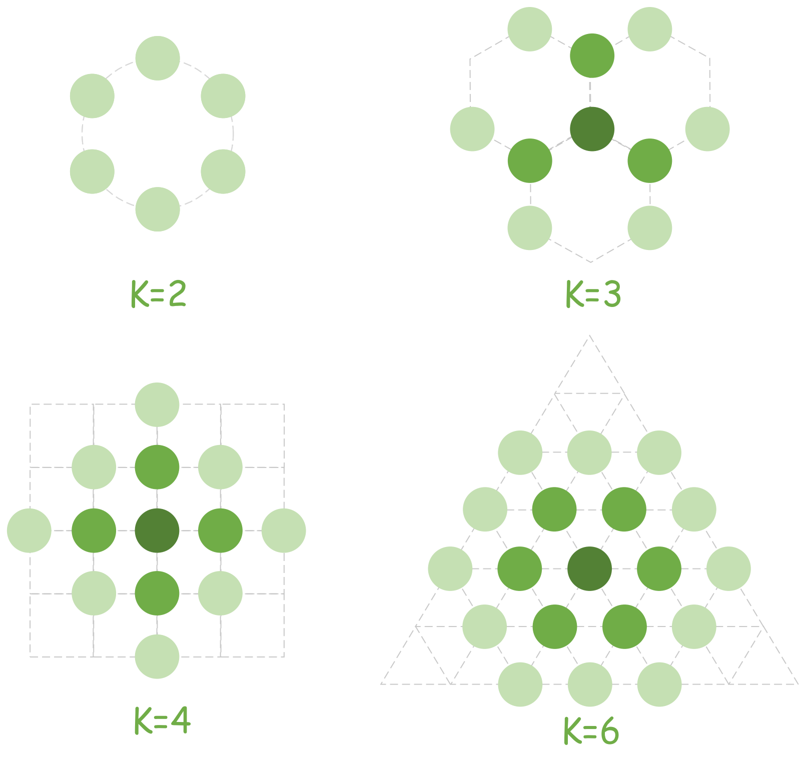


Figure 6: Given a specific value of K, if nodes are uniformly distributed in regular grid patterns such as circles, hexagons, quadrilaterals, and triangles, then each node will have the same in-degree and out-degree. This uniform distribution ensures consistent connectivity across all nodes within these structured arrangements.

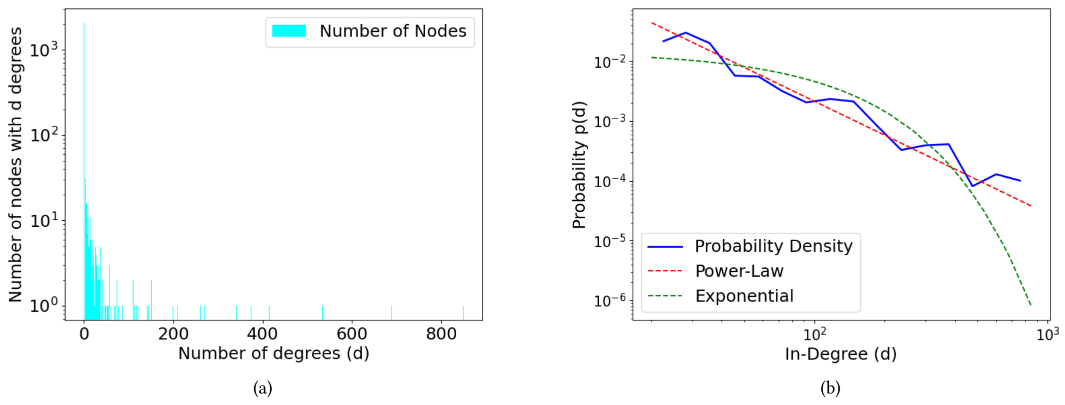


Figure 7: In-degree distribution and fitting curves of the KNN graph using Euclidean distance as metric on the Cora dataset. Note that different from Fig. 4, the y-axis of Fig. 7(a) employs a logarithmic transformation.

[3] Zixing Song, Yifei Zhang, and Irwin King. 2023. Optimal Block-wise Asymmetric Graph Construction for Graph-based Semi-supervised Learning. In *Advances in Neural Information Processing Systems*, A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 71135–71149. https://proceedings.neurips.cc/paper_files/paper/2023/file/e142fd2b70f10db2543c64bca1417de8-Paper-Conference.pdf

[4] Qitian Wu, Chenxiao Yang, Wentao Zhao, Yixuan He, David Wipf, and Junchi Yan. 2023. DIFFormer: Scalable (Graph) Transformers Induced by Energy Constrained

Diffusion. arXiv:2301.09474 [cs.LG]

[5] Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. 2022. NodeFormer: A Scalable Graph Structure Learning Transformer for Node Classification. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 27387–27401. https://proceedings.neurips.cc/paper_files/paper/2022/file/af790b7ae573771689438bbcfc5933fe-Paper-Conference.pdf

[6] Jianxiang Yu, Yuxiang Ren, Chenghua Gong, Jiaqi Tan, Xiang Li, and Xuecang Zhang. 2023. Empower Text-Attributed Graphs Learning with Large Language Models (LLMs). *arXiv* abs/2310.09872 (2023).

[7] Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2022. Learning on Large-scale Text-attributed Graphs via Variational Inference. In *The Eleventh International Conference on Learning Representations*.

[8] Xiong Zhou, Xianming Liu, Hao Yu, Jialiang Wang, Zeke Xie, Junjun Jiang, and Xiangyang Ji. 2024. Variance-enlarged Poisson Learning for Graph-based Semi-Supervised Learning with Extremely Sparse Labeled Data. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=yeeVBMDAwy