LPNL: Scalable Link Prediction with Large Language Models

Baolong Bi[1,3] **Shenghua Liu**[1,3*] **Yiwei Wang**[2] **Lingrui Mei**[1,3] **Xueqi Cheng**[1,3]
[1]CAS Key Laboratory of AI Safety, Institute of Computing Technology, Chinese Academy of Sciences
[2]University of California, Los Angeles
[3]University of Chinese Academy of Sciences
{bibaolong23z,liushenghua,cxq}@ict.ac.cn wangyw.evan@gmail.com meilingrui22@mails.ucas.ac.cn

Abstract

Exploring the application of large language models (LLMs) to graph learning is an emerging endeavor. However, the vast amount of information inherent in large graphs poses significant challenges to graph learning with LLMs. This work focuses on the link prediction task and introduces **LPNL** (**L**ink **P**rediction via **N**atural **L**anguage), a framework based on large language models designed for scalable link prediction on large-scale heterogeneous graphs. We design novel prompts for link prediction that articulate graph details in natural language. We propose a two-stage sampling pipeline to extract crucial information from the graphs, and a divide-and-conquer strategy to control the input tokens within predefined limits, addressing the challenge of overwhelming information. We fine-tune a T5 model based on our self-supervised learning designed for link prediction. Extensive experimental results demonstrate that LPNL outperforms multiple advanced baselines in link prediction tasks on large-scale graphs.

1 Introduction

Heterogeneous graphs (Shi et al., 2016) are commonly employed for modeling complex systems, wherein entities of diverse types interact with each other via various relations. Figure 1 shows the heterogeneous nodes and their relationships sourced from the Open Academic Graph (OAG) (Huang et al., 2020). Link prediction (Zhang and Chen, 2018; Cai et al., 2021) is a fundamental task in graph learning. However, due to the vast quantity of nodes and edges with their complex structure, addressing the link prediction task on large-scale heterogeneous graphs is challenging.

Recently, some research (Fatemi et al., 2023; Ye et al., 2023) has explored the use of large language models (LLMs) in graph learning. A popular paradigm of link prediction on graphs with LLMs is to transform graph problems and structures into description texts, and then feed the texts to LLMs to obtain the predictions. However, it remains under explored that how to perform scalable link prediction on large graphs through LLMs with the input window constraints, which poses serious challenges in capturing distant information and rich semantics. As the number of nodes increases, the text fed into LLMs grows. Consequently, extensive inputs become unfeasible due to token length limitations.

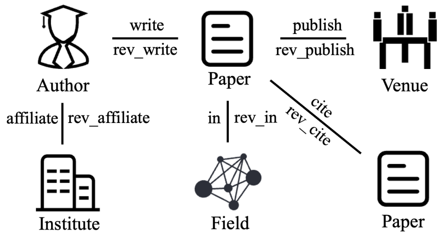


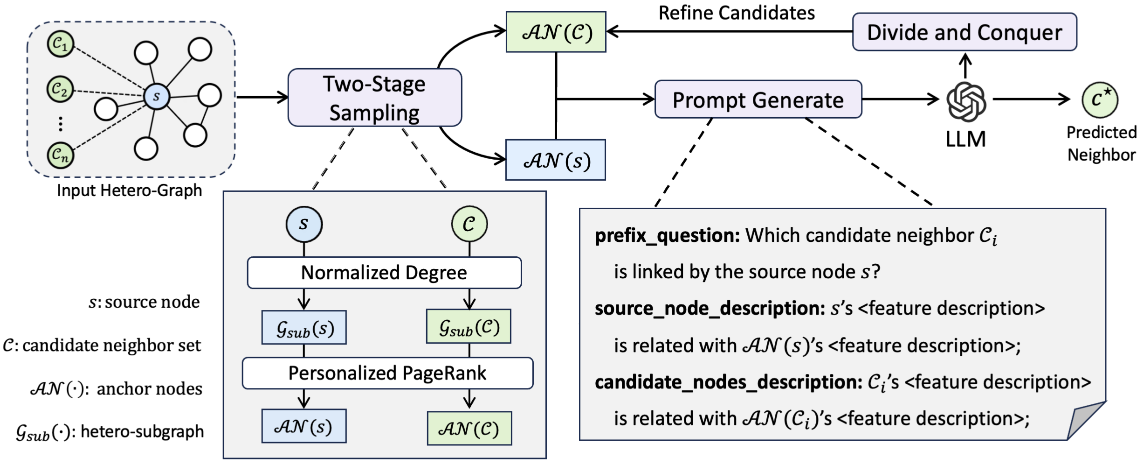
Figure 1: An example of heterogeneous graph

In this work, we explore the scalable link prediction with large language models on large-scale heterogeneous graphs. The key challenges can be described as follows: 1) how to fomulate the prompt template for scalable link prediction task. 2) how to find out crucial information on large graphs, enabling LLMs to capture it within limited inputs. 3) how to address lengthy prompts generated by an excess of candidate neighbors. To tackle the above challenges, we propose **LPNL** (**L**ink **P**rediction via **N**atural **L**anguage), a large language model based framework for scalable link prediction on large-scale graphs. The framework of LPNL is shown in Figure 2.

We design novel prompts for link prediction that articulating graph details in natural language. This involves establishing a selective query prompt template, furnishing a description of the link prediction task, and integrating heterogeneous information concerning the source node and candidate neighbors.

[*] Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2024, pages 3615–3625
August 11-16, 2024 ©2024 Association for Computational Linguistics



prefix_question: Which candidate neighbor $\mathcal{C}_i$ is linked by the source node s?
source_node_description: s's <feature description> is related with $\mathcal{AN}(s)$'s <feature description>;
candidate_nodes_description: $\mathcal{C}_i$'s <feature description> is related with $\mathcal{AN}(\mathcal{C}_i)$'s <feature description>;

Figure 2: The framework of LPNL. For an input heterogeneous graph with link prediction tasks, LPNL consists of three steps: (1) conduct a two-stage sampling on the source node and each candidate neighbor from the original candidate set to acquire anchor nodes. (2) Generate prompts based on these anchor nodes and input them into LLMs for predictions. (3) Refine the candidate set based on prediction results and iteratively apply this divide-and-conquer process to obtain the distinct link prediction result c^*.

In dealing with vast amounts of relevant graph information within large graphs, LPNL selects crucial node information from the graph, ensuring that the model focuses more on them. We design a two-stage sampling pipeline that utilizes normalized degree-based heterogeneous subgraph sampling and personalized pagerank-based ranking. This approach avoids the interference of superfluous contextual information while ensuring compliance with specified token limitations.

With a large number of candidate neighbors, the token length constraints make it challenging to fully describe all candidate neighbor information. To address this issue, we employ a divide-and-conquer method. The original node set is partitioned into multiple sets with smaller size, which are sequentially input into the link prediction pipeline to obtain partial answers. Subsequently, we recursively refine the candidate set to predict the final answer.

We conduct extensive experiments on the OAG and fine-tune the language model T5 (Raffel et al., 2020) based on our self-supervised learning to serve as the backbone model for LPNL on the OAG . The results demonstrate that LPNL significantly outperforms various enhanced GNN-based baselines, achieving an average improvement of 30.52% on Hits@1. Furthermore, through extensive experimentation, LPNL also exhibits remarkable few-shot capability. Unlike traditional models training, LPNL's fine-tuning merely requires simple alignment formatting, enabling swift convergence in predictions. Additionally, experiments demonstrate the model's robust knowledge transferability, maintaining consistent performance across various cross-domain tasks. This further emphasizes that LPNL's self-supervised fine-tuning is not confined to fixed graph labels, it can make direct predictions on different graphs without the need for additional learning.

2 The LPNL Architecture

In this section, we introduce the details of our proposed **L**ink **P**rediction via **N**atural **L**anguage, i.e. **LPNL**, a framework utilizing natural language to solve link prediction task on large-scale heterogeneous graphs. We start with the notation setup, followed by the prompt design, the sampling methods, and our divide-and-conquer and self-supervised strategy with more details.

2.1 Preliminary

Formally, a heterogeneous graph is denoted by $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R}\}$, where $\mathcal{V}$ and $\mathcal{E}$ denote the sets of nodes and edges (links), respectively. Each node $v \in \mathcal{V}$ and each link $e \in \mathcal{E}$ are associated with their mapping function $\phi(v) : v \rightarrow \mathcal{A}$ and $\varphi(e) : e \rightarrow \mathcal{R}$. $\mathcal{A}$ represents the set of node types, and $\mathcal{R}$ represents the set of edge types.

Given a source node s and a set of candidate neighbors $\mathcal{C} = \{c_1, c_2, ..., c_n\}$, satisfying a existed meta-relation $\langle \phi(s), \varphi(e), \phi(c_i) \rangle$ where $e \in \mathcal{E}$ and $c_i \in \mathcal{C}$, a standard link prediction task on heterogeneous graphs aims to predict a candidate neighbor

$c \in \mathcal{C}$ for a source node s with the highest probability of $\langle s, e, c \rangle$.

Finally, let $\mathcal{G}^h_{sub}(v) = \{\mathcal{V}^h_v, \mathcal{E}^h_v, \mathcal{A}^h_v, \mathcal{R}^h_v\}$ denote the h-hop ego-subgraph around v, consisting of h-hop neighbor nodes of v and all interconnecting edges. We also denote $\mathcal{N}^h(v)$ as the set of all neighbor nodes on $\mathcal{G}^h_{sub}(v)$, which means $\mathcal{N}^h(v) = \{v' | v' \in \mathcal{V}^h_v, v' \neq v\}$. Additionally, $\mathcal{AN}^h_k(v)$ is denoted as the sequence of top-k anchor nodes selected from $\mathcal{N}^h(v)$. Note that all the above definitions are heterogeneous.

2.2 Prompt Design for Link Prediction

In order to comprehensively represent the link prediction task along with the essential graph information, we meticulously design a uniform prompt template $\mathcal{T}(\cdot)$ for heterogeneous link prediction. Its fundamental mode involves a selective query, providing both the link prediction problem description and information regarding the source node and candidate neighbors. This prompts the large language models to identify the node most likely to be linked within the candidate set.

First, we define $d(v)$ as the description of node v, which consists of a sequence of textual features of itself and also its top-k anchor nodes:

$$d(v) = \{v : \mathcal{S}_v\} \text{ is related with } \sum_{i=1}^{k} \{v'_i : \mathcal{S}_{v'_i}\} \qquad (1)$$

where $\mathcal{S}_v$ denotes the textual description of node v and v'_i represents the anchor node of node v satisfying $v'_i \in \mathcal{AN}^h_k(v)$.

Subsequently, given a source node s and the set of candidate neighbors $\mathcal{C}$, we formally obtain the link prediction prompt template as follows:

$$\mathcal{T}(s, \mathcal{R}, \mathcal{C}) = q(\mathcal{R}) + d(s) + \sum_{i=1}^{n} d(c_i | c_i \in \mathcal{C}) \qquad (2)$$

where $\mathcal{R}$ is the relation type between the source node and candidate neighbors and n is the number of candidate neighbors. And $q(\mathcal{R})$ represents a link prediction query, e.g., "which $\phi(c)$ is linked by $\phi(u)$?". Notably, in the above equation, the addition operators are redefined as the textual concatenation with separators.

To enhance the capability of the large language models in distinguishing between various types of heterogeneous nodes, we additionally assign distinct type identifiers to the backend of each node. For example, a paper node could be described as "<p>[PA]". Following the formal definition provided above, Figure 3 illustrates a more intuitive prompt example for author disambiguation.

Author Disambiguation Example

prefix_question: Which following candidate author writes the paper p$_1$?

source_node_description: p$_1$: *<paper title>* is related with f$_{25}$: *<field name>*, v$_{13}$: *<journal info>*, p$_{46}$: *<paper title>*, a$_{38}$: *<author info>*, p$_{27}$: *<paper title>...*

candidate_nodes_description: a$_1$: *<author info>* is related with p$_{15}$: *<paper title>...*; a$_2$: ...; a$_3$: ...

Figure 3: The prompt example consists of three components: prefix_question: a selective question; source_node_description: the description of the source node and its corresponding anchor nodes; candidate_nodes_description: the description of candidate neighbors and the anchor nodes corresponding to each candidate neighbor.

Our designed prompts do not explicitly capture the link information between nodes in the graph. Instead, we choose to describe key nodes in textual form based on their order of importance. This decision arises from the complexity of inter-node connections, which often result in redundant contexts (Fatemi et al., 2023), making it challenging for large language models to comprehend. Consequently, there is a risk of LLMs diminishing the emphasis on node features, which are pivotal for our tasks. Nonetheless, the links among heterogeneous nodes remain crucial as they reflect their relationships and node significance. In following Sec.2.3, we introduce a two-stage sampling approach to leverage structural information, prioritizing critical nodes and thereby enhancing the description of graph information.

2.3 Two-Stage Sampling

In the previous subsection, we designed the unified prompt template for link prediction. However, as graph data becomes more complex, resembling the real world, employing a single prompt engineering approach becomes challenging in addressing practical application problems. Firstly, in large-scale graphs, attempting to describe the node information of v using all h-hop neighbors, i.e. $k = |\mathcal{V}_v^h|$ formally, as shown in Eq.(1), leads to an uncontrollable prompt length. Anothor issue arises due to substantial variations in the degrees of different node types. For example, the number of nodes in the h-hop subgraph around a paper node is significantly smaller than that around a field-type node. The two problems pose significant challenges to the input and contextual comprehension of LLMs.

In this work, we provide a two-stage sampling pipeline. The first stage aims to sample subgraphs $\mathcal{G}_{sub}^h(v)$ based on normalized degree from large-scale heterogeneous graphs while mitigating sampling bias caused by heterogeneous types. Subsequently, we obtain the top-k anchor nodes sequence $\mathcal{AN}_k^h(v)$ through the second stage sampling with personalized pagerank to generate $d(v)$ in Eq.(1). The further details are as follows.

Normalized Degree based Sampling Inspired by previous studies (Hu et al., 2020; Leskovec and Faloutsos, 2006), we adopt a strategy for sampling heterogeneous subgraphs based on normalized degree. Specifically, this approach specifies the sampling probability of each hop's neighbors as their normalized degree. The normalized degree is defined as the node's degree normalized among all nodes of the same type in the same layer. Therefore, for the l-th layer subgraph sampling around central node s, the sampling probability of node v can be described as follows:

$$prob_s^l(v) = \frac{deg(v)^2}{\sum deg(u)^2} \quad (3)$$

where node u represents the neighbor node at the l-th layer within the subgraph, satisfying $u \in \mathcal{V}_s^h \setminus \mathcal{V}_s^{h-1}$ and $\phi(v) = \phi(u)$.

The normalized degree based sampling in our first stage ensures that differences between node types are not ignored, preventing bias against certain node types (e.g., nodes with higher degrees are not indiscriminately considered more important). This approach maintains a similar number of different types of nodes in the subgraphs, thereby preserving richer semantic information. Furthermore, previous studies have demonstrated that leveraging up to 3-hop connectivity is effective for achieving excellent performance (Kipf and Welling, 2016; Veličković et al., 2017; Hamilton et al., 2017). However, extending the information beyond 3-hop generally has a marginal impact on improvement and, in some cases, may even result in negative effects (Cai and Wang, 2020; Zhang et al., 2021). Therefore, we set the maximum value for multi-hop to 2-hop or 3-hop in our two-stage sampling approach.

Sampling with Personalized PageRank Through the sampling in the first stage, the heterogeneous subgraphs we obtain eliminate biases between different types, allowing all types of nodes to be compared regarding their importance on an equal footing. In the second stage, we directly compute the importance of all neighbor nodes within the subgraph $\mathcal{G}_{sub}^h(v)$ for the source node s using Personalized PageRank (PPR) (Bojchevski et al., 2020; Vattani et al., 2011). We then obtain the PPR vector $\vec{\pi_s}$ for the source node s by iteratively updating the following:

$$\vec{\pi_s} = \alpha * \vec{e_s} + (1-\alpha) * A^\top D^{-1} \vec{\pi_s} \quad (4)$$

where α denotes the damping factor, A stands for the adjacency matrix, D^{-1} denotes the diagonal degree matrix and $\vec{\pi_s}$ signifies the unit vector.

This work employs a queue-based implementation of the equivalent random walk (Spitzer, 2013; Wu et al., 2021) to approximate PPR. Subsequently, the top-k anchor nodes sequence $\mathcal{AN}_k^h(s)$ is obtained based on the ranking derived from PPR, which characterizes the top-k neighbor nodes that are most critical for the source node s within the whole hetero-graph.

The two-stage sampling restricts the generated link prediction prompt length to suit LLMs inputs while maximizing the retention of crucial neighborhood information pertaining to the target node within the subgraphs. It also makes use of the structural information on the graph, so that the generated anchor nodes $\mathcal{AN}_k^h(s)$ can be seen as a hub converting the graph structure into textual descriptions. This enables our prompts generated in Sec.2.2 to encompass not only node features but also implicit structural information.

2.4 Divide-and-Conquer Prediction

While the sampling pipeline addresses the potential issue of prompts length caused by Eq.(1), a careful observation of Eq.(2) reveals that an excessive number of candidate neighbors in link prediction, denoted as $|\mathcal{C}|$, also makes the prompt length uncontrollable. Especially in large-scale graphs, the high number of candidate neighbors poses a challenge in describing all of them within a single LLM's input window. For instance, in a link prediction task with 100 candidate neighbors, each node requires an average of approximately 200 tokens in

Dataset	#nodes	#edges	#papers	#authors	#fields	#venues	#institutes	#P-A	#P-F	#P-V	#A-I	#P-P
CS	11,918,983	107,263,811	5,597,605	5,985,759	119,537	27,433	16,931	15,571,614	47,462,559	5,597,606	7,190,480	31,441,552
Mater	4,552,941	42,161,581	2,442,235	2,005,362	79,305	15,141	10,898	5,582,765	19,119,947	2,442,235	2,005,362	13,011,272
Engin	5,191,920	36,146,719	3,239,504	1,819,100	99,444	19,867	14,005	3,741,135	22,498,822	3,239,504	1,819,100	4,848,158
Chem	12,158,967	159,537,437	7,193,321	4,748,812	183,782	19,142	13,910	16,414,176	57,162,528	7,193,321	4,748,812	74,018,600

Table 1: OAG statistics.

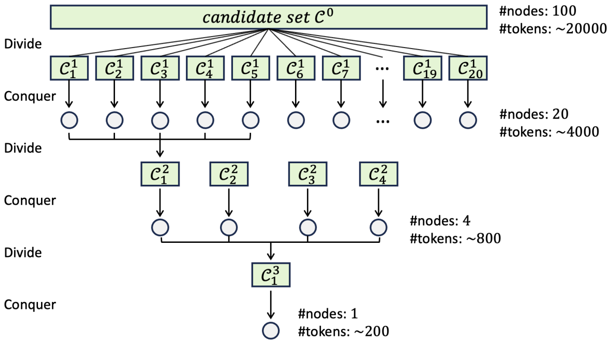

candidate set $\mathcal{C}^0$
#nodes: 100
#tokens: ~20000
Divide
$\mathcal{C}_1^1$ $\mathcal{C}_2^1$ $\mathcal{C}_3^1$ $\mathcal{C}_4^1$ $\mathcal{C}_5^1$ $\mathcal{C}_6^1$ $\mathcal{C}_7^1$ ... $\mathcal{C}_{19}^1$ $\mathcal{C}_{20}^1$
Conquer
...
#nodes: 20
#tokens: ~4000
Divide
$\mathcal{C}_1^2$ $\mathcal{C}_2^2$ $\mathcal{C}_3^2$ $\mathcal{C}_4^2$
Conquer
#nodes: 4
#tokens: ~800
Divide
$\mathcal{C}_1^3$
Conquer
#nodes: 1
#tokens: ~200

Figure 4: For a link prediction task involving 100 candidate neighbors, we set the candidate length limit L to 5. The candidate neighbors can be divided into 20 sets, followed by three rounds of divide-and-conquer. This process ultimately yields a unique prediction result.

the prompt for description. This results in a total of 20,000 tokens needed to describe all candidate neighbors, far exceeding the maximum token limit for a usual LLM's input window. Furthermore, the excessive number of candidate neighbors leads to redundant contexts, making it challenging for the LLMs to comprehend the input text.

LPNL avoids the aforementioned token overload by employing a divide-and-conquer strategy. Figure 4 provides an intuitive example of the divide-and-conquer prediction, allowing us to observe the descent of candidate neighbors and prompt tokens throughout the process. We set a length limit L for the candidate set, ensuring that the length of all processed candidate sets does not exceed L. We represent $\mathcal{C}_j^i$ as the j-th candidate set of the i-th divide-and-conquer round. Specifically, for an original candidate set of length $|\mathcal{C}^0|$ where $|\mathcal{C}^0| > L$, we randomly divide it into m sub-candidate sets, ensuring $m = \lceil \frac{|\mathcal{C}^0|}{L} \rceil$. This results in sets denoted as $\mathcal{C}_1^1, \mathcal{C}_2^1, ..., \mathcal{C}_m^1$, with the constraint that $max(|\mathcal{C}_1^1|, |\mathcal{C}_2^1|, ..., |\mathcal{C}_m^1|) \leq L$.

As illustrated in Figure 1, by employing the fine-tuned large language models to predict the candidate neighbor of the source node with the maximum link probability for each sub-candidate set, we can subsequently eliminate low-probability candidate neighbors. And the process generates new candidate sets based on the predicted results by refining the candidate sets. Specifically, for the candidate sets $C_{j+1}^i, C_{j+2}^i, ..., C_{j+k}^i$, a new candidate set $C_{k'}^{i+1}$ is generated in the following round based on their prediction results. The values of k and k' are determined based on the order of generation, ensuring that the condition $k \leq L$ is met. Following this divide-and-conquer process by refining candidate sets and making predictions, ultimately, we can obtain a unique prediction answer for the entire original candidate set $\mathcal{C}^0$.

2.5 Self-Supervised Fine-tuning

As a more relevant graph structure, large-scale graphs lack labelled data. LPNL uses self-supervised learning for large language model fine-tuning. During the end-to-end prompt fine-tuning, it automatically constructs a candidate set containing ground truth, aligned with downstream prediction formats. The ground truth is used as the correct answer for link prediction. To ensure training correctness, the ground truth appears randomly within the candidate neighbor sequence. Notably, during the heterogeneous subgraph sampling process in Sec.2.3, the edges between the ground truth and the source node are masked. Because the self-supervised fine-tuning does not require training labels provided by graph tasks, a fine-tuned LPNL model can make direct predictions on different graphs without the need of extra tuning.

3 Experiments

3.1 Experiment Settings

Models We fine-tune T5-base model (Chung et al., 2022) with a 1024 input window constraint as the backbone language model for our LPNL. The numbers of sampling hops $h = 2$, top anchor nodes sequence $k = 50$, and candidate length limit $L = 3$ are used for all following experiments.

Datasets We conducted all experiments on the OAG, known as one of the largest publicly available heterogeneous graphs, comprising 178 million

Dataset	Metric	GraphSage	HGT	RGCN	GCN	GAT	LPNL	Δ
CS	NDCG	.814±.025	.847±.042	.843±.056	.887±.031	.911±.033	**.985±.008**	↑ 8.12%
	MRR	.640±.045	.712±.024	.685±.056	.727±.032	.797±.051	**.939±.018**	↑ 17.81%
	Hits@1	.469±.012	.562±.022	.532±.056	.568±.011	.686±.014	**.894±.004**	↑ 30.32%
Mater	NDCG	.765±.017	.841±.034	.854±.042	.818±.016	.897±.065	**.954±.014**	↑ 6.35%
	MRR	.519±.052	.643±.031	.665±.027	.667±.036	.747±.058	**.881±.011**	↑ 17.93%
	Hits@1	.278±.016	.447±.019	.476±.028	.524±.032	.597±.018	**.809±.007**	↑ 35.51%
Engin	NDCG	.798±.021	.876±.022	.874±.061	.912±.041	.913±.037	**.977±.017**	↑ 7.01%
	MRR	.570±.027	.691±.041	.699±.034	.747±.023	.769±.041	**.917±.017**	↑ 16.14%
	Hits@1	.342±.023	.506±.018	.523±.056	.583±.021	.624±.011	**.858±.012**	↑ 37.50%
Chem	NDCG	.821±.015	.863±.015	.835±.036	.893±.017	.899±.023	**.955±.018**	↑ 6.23%
	MRR	.649±.034	.724±.027	.678±.031	.749±.023	.780±.029	**.872±.038**	↑ 11.79%
	Hits@1	.485±.024	.523±.031	.530±.016	.609±.020	.667±.022	**.792±.007**	↑ 18.74%

Table 2: Experimental results of different methods over the four datasets.

nodes and 2.236 billion edges. It includes five types of nodes (denoted as papers (P), authors (A), venues (V), institutes (I) and fields (F)) and their interrelations. In our specific experiments, we utilized four representative domain-specific subgraphs from OAG: Computer Science (CS), Material Science (Mater), Engineering (Engin) and Chemistry (Chem) (Jiang et al., 2021). The graph statistics are listed in Table 1. We partition each dataset into fine-tuning, validation, and test sets based on distinct time periods. Specifically, in the OAG dataset, papers are published between 1900 and 2019. Consequently, we utilize publications preceding 2015 for fine-tuning, data from 2015 to 2016 for validation, and information from 2016 onwards for testing.

Task We consider real-world link prediction tasks to evaluate the performance of our LPNL, specifically, author name disambiguation (Ferreira et al., 2012). Author name disambiguation is a fundamental challenge for curating academic publication and author information, as duplicated names are common. The objective is to predict the true author who has a genuine link with a given paper among all authors with the same name.

Baselines We select a series of supervised baselines, all of which are advanced graph neural network models. These include GCN (Li et al., 2018), GraphSage (Hamilton et al., 2017) and GAT (Veličković et al., 2017), designed for homogeneous graphs, as well as RGCN (Schlichtkrull et al., 2018) and HGT (Hu et al., 2020), tailored for heterogeneous graphs.

3.2 Overall Performance

In this experiment, we compare the T5 model as the backbone version of LPNL to advanced GNN based baseline models across the four domain-specific subgraphs. We fine-tune the model separately across various subgraphs and evaluate the performance of the models in link prediction. The experimental results of the proposed method and baselines are summarized in Table 2. All experiments for the author name disambiguation task over all datasets are evaluated in terms of NDCG, MRR and Hits@1 (Li, 2022; Liu et al., 2009).

The results show that in terms of all three metrics, the proposed LPNL significantly and consistently outperforms all baselines for all tasks on all datasets. Overall, our LPNL consistently yields the best performance among all methods, leading to an average improvement of 6.93%, 15.92% and 30.52%, compared to the second best baseline method. Surprisingly, LPNL exhibited significant improvements in all settings, particularly in the Hit@1 metric. The substantial leap in achieving correct predictions with just a single attempt holds significant implications for practical applications. These improvements over GNNs indicate the efficacy of our proposed LPNL in enabling large language models to comprehend link prediction tasks within complex graphs and large language models have tremendous potential in addressing graph-related problems.

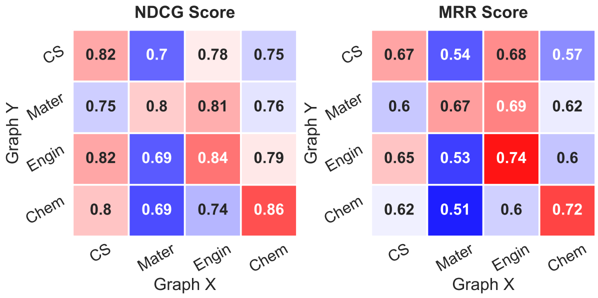


Figure 5: Cross-domain transfer results.

3.3 Cross-Domain Knowledge Transfer

To explore the generalization capabilities of LPNL, we set up experiments for cross-domain knowledge transfer. Specifically, we fine-tune the T5 model using LPNL on a graph corresponding to one domain and subsequently conducted testing on subgraphs from other domains. The experimental outcomes, visualized in Figure 5 as a heatmap, reveal that in most instances, the model exhibits optimal performance when fine-tuned within its original domain. Surprisingly, the models fine-tuned on other domains also demonstrate remarkably strong performance, often closely matching or even surpassing the best performance achieved by fine-tuning within the original domain (e.g., Mater-Engin). This highlights the robust knowledge transferability of our approach, which means it can make direct predictions on different graphs without the need for additional learning.

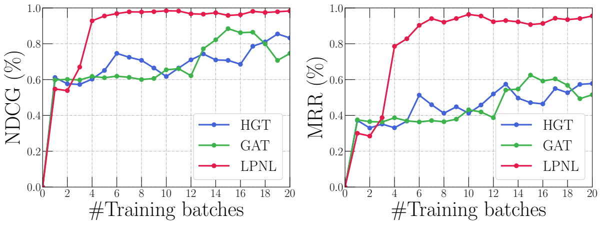


Figure 6: LPNL converges fast in few-shot learning compared to GNNs.

3.4 Few-Shot Learning

The extensive pretraining of large language models across various natural language tasks has endowed them with robust reasoning and generalization capabilities. In contrast to traditional GNN models, they require minimal training samples to converge and exhibit superior performance. We further investigates the few-shot learning capabilities of LPNL by comparing it with the top-performing homogeneous GNN and heterogeneous GNN in terms of overall performance. We configure the evaluate results to be printed every 1 batch, with each batch consisting of 50 link prediction tasks. We compare the few-shot results for the first 20 batches. The results in Table 6 demonstrate that our LPNL swiftly converges with minimal sample fine-tuning, displaying comparable performance to the best fine-tuning outcomes. This showcases the portability of large language models in addressing graph-related tasks.

Method	NDCG	MRR	Hits@1
LPNL	**97.86**	**94.37**	**89.47**
w/o Graph Info	68.98	50.51	35.97
w/o Stage 1	76.31	57.29	41.83
w/o Stage 2	87.67	70.67	53.33

Table 3: Ablation study results of sampling methods.

3.5 Ablation Study

We conduct an ablation study on CS dataset to evaluate the effectiveness of our approach in employing large language models combined with graph knowledge strategies. We compare the performance among different versions of sampling methods: the standard LPNL, a version without any graph information, and another two sampling versions, each independently utilizing distinct stages. As illustrated in Table 3, the model's performance significantly diminishes when graph information is excluded. Furthermore, the performance of the versions without stage 1 and stage 2 shows a notable gap compared to LPNL. It indicates that employing our designed two-stage sampling pipeline enables LPNL to capture crucial information within the graph after balanced heterogeneous sampling, resulting in improved predictive outcomes.

Hop	NDCG	MRR	Hits@1
2-hop	**97.86**	**94.37**	**89.47**
1-hop	94.15	91.56	85.09

Table 4: Ablation study results of multi-hop sampling.

In our experiments, two critical operations contributing significantly to the outstanding performance of LPNL in link prediction are 2-hop and anchor nodes, which provide essential information to the LLMs. To assess the impact of these two key components on model performance, we conducted another ablation experiments, and the results are presented in Table 4 and 5. It shows that incorporating multi-hop and more anchor nodes information can both enhance the LPNL's performance. However, further experiments indicate that increasing the number of hops and anchor nodes beyond a certain threshold does not lead to significant performance improvement. On the contrary, it may result in additional costs without notable benefits.

#Anchor Nodes	NDCG	MRR	Hits@1
Top-30	92.86	76.48	62.05
Top-50	97.86	**94.37**	**89.47**
Top-70	**98.06**	93.84	88.69

Table 5: Ablation study results of top-k anchor nodes

4 Related Work

Graph Representation Learning Based on GNNs Graph Neural Networks (GNNs) are the forefront of graph representation learning methods and have gained significant popularity across a range of graph-related tasks (Wu et al., 2020; Zhou et al., 2020). In these tasks, such as node classification and link prediction, GNNs-based approaches usually preprocess the corresponding text by a language model and encode the resulting embedding as node features. The final node representation is then obtained by aggregating the neighborhood features through spectral methods (Bruna et al., 2013; Defferrard et al., 2016) and message passing (Abu-El-Haija et al., 2019; Hamilton et al., 2017; Schlichtkrull et al., 2018). Besides, some studies have attempted to propose the GNNs architectures on heterogeneous graphs (Dong et al., 2020; Wang et al., 2019; Hu et al., 2020). Notably, influenced by large language models, recent studies (Sun et al., 2023; Huang et al., 2023)have explored the potential of GNNs in prompt learning. And there have also been attempts (Ioannidis et al., 2022; Zhao et al., 2022) to explore collaborative training between Language Models and GNNs.

Large Language Models with Graph Knowledge The emergence of large language models (LLMs) has propelled natural language processing (NLP) to new heights (Qiu et al., 2020; OpenAI, 2022, 2023; Touvron et al., 2023). LLMs have found widespread applications in many scenarios (Bi et al., 2024a,b; Mei et al., 2024; Ni et al., 2023, 2024; Fan et al., 2024). Models like BERT (Devlin et al., 2018) and T5 (Raffel et al., 2020) demonstrate excellent performance in a wide range of downstream tasks, such as text classification and question answering. Besides, some works (Zhang et al., 2019; Liu et al., 2022, 2020)attempts to inject external graph knowledge into LLMs, thus enabling LLMs to gain the ability to solve problems on graphs. Recently, due to the powerful inferential capabilities of large language models, a burgeoning body of work (Fatemi et al., 2023; Ye et al., 2023; Liu et al., 2023) attempt to utilize natural language descriptions of graph features, employing generated prompts to instruct large language models in addressing various problems on graphs.

5 Discussion

From our experiments, we found that describing graphs using natural language does not follow the principle of "more information is better". Sampling more nodes can introduce additional information, but it may lead to information redundancy, resulting in a decline in the inferential capabilities of large language models. Therefore, the key lies in the setting of the sampling and divide-and-conquer length limits, which should align with the input window size and inferential capabilities of the large language models. While designing prompts, we observe that complex relationships between nodes are challenging to articulate in text, especially in large or dense graphs, potentially leading to redundant contexts. LPNL leverages structural information during the sampling phase and, in the prompt generation, only conveys information about the sampled nodes. This approach aims to minimize context redundancy while maximizing the utilization of graph information.

6 Conclusion

In this paper, we explore, for the first time, the application of large language models to address the link prediction task on large-scale heterogeneous graphs. We introduce LPNL, a large language models based framework for scalable link prediction on large-scale graphs. We design specific prompt templates for the link prediction task and generate the prompts based on anchor nodes obtained through a two-stage sampling approach. These prompts are then input to the large language models for predictions. To tackle the token overload issue arising from an excessive number of candidate neighbors, we employ a divide-and-conquer strategy. Empirical evaluations demonstrate that LPNL achieves significant improvements compared to GNN baselines, showcasing its robust capability in cross-domain knowledge transfer and few-shot learning scenarios.

Limitations

Some efforts in solving graph-related problems using LLMs involve supervised fine-tuning, resulting in limited ability for knowledge transfer. Although LPNL supports unsupervised learning without the need for labels, it is currently confined to link prediction tasks and has not been applied to a broader spectrum of graph-related tasks. LPNL has not yet explored larger parameter scales for large language models and their zero-shot potentials, which could provide increased input window sizes and enhanced inferential capabilities. Integrating our approach with other graph tasks and larger language models holds the potential to significantly improve predictive capabilities.

Ethics Statement

Ethical considerations are of utmost importance in our research endeavors. In this paper, we conscientiously adhere to ethical principles by exclusively utilizing open-source datasets and employing models that are either open-source or widely recognized in the scientific community. Moreover, our proposed method is designed to ensure that the model does not produce any harmful or misleading information. We are committed to upholding ethical standards throughout the research process, prioritizing transparency, and promoting the responsible use of technology for the betterment of society.

Acknowledgements

This paper is partially supported by the National Science Foundation of China under Grant No. U21B2046 and 6237075198, and National Key R&D Program of China (No. 2023YFC3305303).

References

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. 2019. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, pages 21–29. PMLR.

Baolong Bi, Shenghua Liu, Lingrui Mei, Yiwei Wang, Pengliang Ji, and Xueqi Cheng. 2024a. Decoding by contrasting knowledge: Enhancing llms' confidence on edited facts. *arXiv preprint arXiv:2405.11613*.

Baolong Bi, Shenghua Liu, Yiwei Wang, Lingrui Mei, and Xueqi Cheng. 2024b. Is factuality decoding a free lunch for llms? evaluation on knowledge editing benchmark. *arXiv preprint arXiv:2404.00216*.

Aleksandar Bojchevski, Johannes Gasteiger, Bryan Perozzi, Amol Kapoor, Martin Blais, Benedek Rózemberczki, Michal Lukasik, and Stephan Günnemann. 2020. Scaling graph neural networks with approximate pagerank. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2464–2473.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Chen Cai and Yusu Wang. 2020. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*.

Lei Cai, Jundong Li, Jie Wang, and Shuiwang Ji. 2021. Line graph neural networks for link prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5103–5113.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yuxiao Dong, Ziniu Hu, Kuansan Wang, Yizhou Sun, and Jie Tang. 2020. Heterogeneous network representation learning. In *IJCAI*, volume 20, pages 4861–4867.

Run-Ze Fan, Xuefeng Li, Haoyang Zou, Junlong Li, Shwai He, Ethan Chern, Jiewen Hu, and Pengfei Liu. 2024. Reformatted alignment. *arXiv preprint arXiv:2402.12219*.

Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2023. Talk like a graph: Encoding graphs for large language models. *arXiv preprint arXiv:2310.04560*.

Anderson A Ferreira, Marcos André Gonçalves, and Alberto HF Laender. 2012. A brief survey of automatic methods for author name disambiguation. *Acm Sigmod Record*, 41(2):15–26.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pages 2704–2710.

Han Huang, Hongyu Wang, and Xiaoguang Wang. 2020. An analysis framework of research frontiers based on the large-scale open academic graph. *Proceedings of the Association for Information Science and Technology*, 57(1):e307.

Qian Huang, Hongyu Ren, Peng Chen, Gregor Kržmanc, Daniel Zeng, Percy Liang, and Jure Leskovec. 2023. Prodigy: Enabling in-context learning over graphs. *arXiv preprint arXiv:2305.12600*.

Vassilis N Ioannidis, Xiang Song, Da Zheng, Houyu Zhang, Jun Ma, Yi Xu, Belinda Zeng, Trishul Chilimbi, and George Karypis. 2022. Efficient and effective training of language and graph neural network models. *arXiv preprint arXiv:2206.10781*.

Xunqiang Jiang, Tianrui Jia, Yuan Fang, Chuan Shi, Zhe Lin, and Hui Wang. 2021. Pre-training on large-scale heterogeneous graph. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 756–766.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Jure Leskovec and Christos Faloutsos. 2006. Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636.

Hang Li. 2022. *Learning to rank for information retrieval and natural language processing*. Springer Nature.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. 2023. One for all: Towards training one graph model for all classification tasks. *arXiv preprint arXiv:2310.00149*.

Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908.

Xiao Liu, Da Yin, Jingnan Zheng, Xingjian Zhang, Peng Zhang, Hongxia Yang, Yuxiao Dong, and Jie Tang. 2022. Oag-bert: Towards a unified backbone language model for academic knowledge services. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3418–3428.

Lingrui Mei, Shenghua Liu, Yiwei Wang, Baolong Bi, and Xueqi Cheng. 2024. Slang: New concept comprehension of large language models.

Shiyu Ni, Keping Bi, Jiafeng Guo, and Xueqi Cheng. 2023. A comparative study of training objectives for clarification facet generation. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 1–10.

Shiyu Ni, Keping Bi, Jiafeng Guo, and Xueqi Cheng. 2024. When do llms need retrieval augmentation? mitigating llms' overconfidence helps retrieval augmentation. *arXiv preprint arXiv:2402.11457*.

OpenAI. 2022. large-scale generative pre-training model for conversation. *OpenAI blog*.

OpenAI. 2023. Gpt-4 technical report.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, pages 593–607. Springer.

Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2016. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):17–37.

Frank Spitzer. 2013. *Principles of random walk*, volume 34. Springer Science & Business Media.

Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. 2023. All in one: Multi-task prompting for graph neural networks.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton

Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Andrea Vattani, Deepayan Chakrabarti, and Maxim Gurevich. 2011. Preserving personalized pagerank in subgraphs. In *ICML*, volume 11, pages 793–800.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032.

Hao Wu, Junhao Gan, Zhewei Wei, and Rui Zhang. 2021. Unifying the global and local approaches: an efficient power iteration with forward push. In *Proceedings of the 2021 International Conference on Management of Data*, pages 1996–2008.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24.

Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. 2023. Natural language is all a graph needs. *arXiv preprint arXiv:2308.07134*.

Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31.

Wentao Zhang, Zeang Sheng, Yuezihan Jiang, Yikuan Xia, Jun Gao, Zhi Yang, and Bin Cui. 2021. Evaluating deep graph neural networks. *arXiv preprint arXiv:2108.00955*.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*.

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. *arXiv preprint arXiv:2210.14709*.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81.