Killing Two Birds with One Stone: Cross-modal Reinforced Prompting for Graph and Language Tasks

Wenyuan Jiang*
School of Computer Science and Technology, University of Science and Technology of China
Hefei, China
johnrosenberg12138@gmail.com

Wenwei Wu*
Thrust of Financial Technology, The Hong Kong University of Science and Technology (Guangzhou), China
wenweiwu@hkust-gz.edu.cn

Le Zhang*
Baidu Research,
Baidu Inc.
Beijing, China
zhangle0202@gmail.com

Zixuan Yuan†
Jian Xiang
Thrust of Financial Technology, The Hong Kong University of Science and Technology (Guangzhou), China
zixuanyuan@hkust-gz.edu.cn
jian.xiang@rutgers.edu

Jingbo Zhou
Baidu Research,
Baidu Inc.
Beijing, China
zhoujingbo@baidu.com

Hui Xiong†
Thrust of Artificial Intelligence, The Hong Kong University of Science and Technology (Guangzhou), China
Department of Computer Science and Engineering, The Hong Kong University of Science and Technology
Hong Kong SAR, China
xionghui@ust.hk

ABSTRACT

In recent years, Graph Neural Networks (GNNs) and Large Language Models (LLMs) have exhibited remarkable capability in addressing different graph learning and natural language tasks, respectively. Motivated by this, integrating LLMs with GNNs has been increasingly studied to acquire transferable knowledge across modalities, which leads to improved empirical performance in language and graph domains. However, existing studies mainly focused on a single-domain scenario by designing complicated integration techniques to manage multimodal data effectively. Therefore, a concise and generic learning framework for multi-domain tasks, i.e., graph and language domains, is highly desired yet remains underexploited due to two major challenges. First, the language corpus of downstream tasks differs significantly from graph data, making it hard to bridge the knowledge gap between modalities. Second, not all knowledge demonstrates immediate benefits for downstream tasks, potentially introducing disruptive noise to context-sensitive models like LLMs. To tackle these challenges, we propose a novel plug-and-play framework for incorporating a lightweight cross-domain prompting method into both language and graph learning tasks. Specifically, we first convert the textual input into a domain-scalable prompt, which not only preserves the semantic and logical contents of the textual input, but also highlights related graph information as external knowledge for different domains. Then, we develop a reinforcement learning-based method to learn the optimal edge selection strategy for useful knowledge extraction, which profoundly sharpens the multi-domain model capabilities. In addition, we introduce a joint multi-view optimization module to regularize agent-level collaborative learning across two domains. Finally, extensive empirical justifications over 23 public and synthetic datasets demonstrate that our approach can be applied to diverse multi-domain tasks more accurately, robustly, and reasonably, and improve the performances of the state-of-the-art graph and language models in different learning paradigms.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; *Multi-agent planning*.

KEYWORDS

Prompt Learning; Graph Neural Networks; Reinforcement Learning; Large Language Models

ACM Reference Format:
Wenyuan Jiang, Wenwei Wu, Le Zhang, Zixuan Yuan, Jian Xiang, Jingbo Zhou, and Hui Xiong. 2024. Killing Two Birds with One Stone: Cross-modal Reinforced Prompting for Graph and Language Tasks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671742

*These authors contributed equally to this work.
†Corresponding authors.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3671742

1 INTRODUCTION

Graph Neural Networks (GNNs) are capable of capturing topological information through message propagation and aggregation mechanisms, making them robust and effective at graph learning tasks [62, 63, 76]. Large Language Models (LLMs), on the other hand, have shown remarkable effectiveness across a broad range of

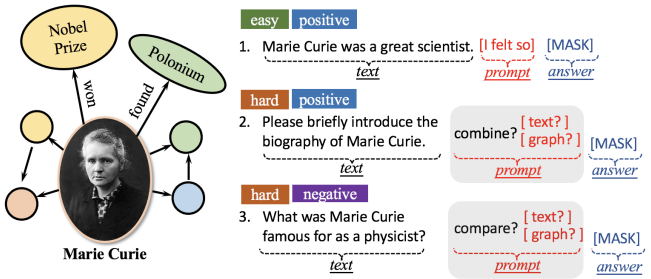
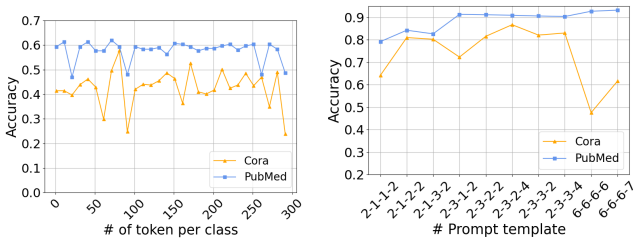


Figure 1: Challenges of creating a cross-modal prompt for language tasks.



(a) Graph prompt. (b) Language prompt.

Figure 2: Node classification accuracies on Cora and PubMed datasets of using (a) graph prompt [45] or (b) language prompt [69]. In case (a), we consider varying numbers of prompt tokens allocated to each class on both datasets. A fluctuation in accuracy can be observed with the change in the number of tokens, indicating that model performance is sensitive to the prompt structure. In case (b), we evaluate the model performance across 10 different prompt templates represented by unique four-digit codes. Similar fluctuation in accuracy suggests the changes in prompt templates greatly impact the classification results.

Natural Language Processing (NLP) applications [3, 5, 32], primarily due to their versatility in handling diverse tasks through zero or few-shot learning approaches. Therefore, the AI communities have witnessed growing attention to integrating versatile LLMs with GNNs through multi-modal capabilities [18, 38, 53, 60, 65]. Such integration is naturally motivated by a dual advantage. First, LLMs alleviate GNNs' reliance on the expensive annotation of nodes or graphs for effective model training [53, 65]. Second, GNNs provide LLMs with informative structural and textual-attributed patterns to elevate their performance in unseen scenarios [38, 60].

Along this line, several integration approaches have been proposed to conduct knowledge fusion engineering [8, 34, 40, 42, 44, 52, 54, 57]: developing either advanced neural units (e.g., adapter module [8, 57], attention blocks [44, 54]) or hand-crafted search algorithms [40, 42] to exploit modal counterpart for rich feature-level or task-level knowledge. Despite their efficacy in various tasks, these approaches usually require massive labeled training data, which is arduously expensive and sometimes inaccessible in practical settings [45]. To address these issues, recent studies resort to the "pre-train, fine-tune" [43, 45] that leverages substantial corpus of unlabeled structure as pretext tasks (e.g., edge prediction for GNNs, masked language modeling for LLMs) to pre-train models and fine-tune them to perform downstream tasks. However, this paradigm is largely limited by an ideal assumption that there exists a small training objective gap between the constructed pretext and dedicated downstream tasks. Fortunately, the emergence of the prompt-based learning [53, 69] provides an effective alignment between the pretext tasks and downstream tasks by using a prompt to modify the input graph or textual data. For example, consider the sentiment analysis task represented by the statement "Marie Curie was a great scientist," depicted in Figure 1. By employing a semantic textual prompt template, e.g., "I felt so [MASK]," the original task is transformed into word prediction, which is achieved by rephrasing the input to "Marie Curie was a great scientist. I felt so [MASK]". In this way, LLMs can naturally predict '[MASK]' as 'admired' rather than 'hated' without additional parameter optimization, as the model can elicit relevant knowledge from similar pre-training tasks via prompts to smoothly answer this query.

Based on the merits of language or graph prompts being applied in a single domain, in this paper, we extend the prompt-based learning to multi-domain scenarios, where multimodal data are potentially beneficial to different domains [27, 66]. However, developing a suitable and efficient prompting strategy to transfer multimodal knowledge across domains is a non-trivial undertaking due to the following challenges. (1) **Bridging knowledge gap between modalities.** The language corpus of downstream tasks is often relevant to graph data, but they are stored in two distinct knowledge representations [34]. While some efforts have been made to unify the multimodal knowledge into a single modality through prompts [9, 30, 71], it remains unclear how to build robust semantic alignment between different modalities, which is critical to effectively addressing complicated tasks. For instance, as shown in Figure 1, textual questions like "Please briefly introduce the biography of Marie Curie" usually ask for a high-level summary for which we need to find answers from low-level knowledge edges in a graph. A promising solution is to establish an information venue that ensures a compatible message-sharing mechanism between graph and text, such as OntoPrompt [68] that adopts knowledge graph ontology to convert low-level structured knowledge to high-level textual prompts. However, such design highly requires both expert knowledge and tedious manual trials to make prompt templates applicable to every new downstream application. (2) **Incorporating noisy knowledge into context-sensitive models.** It has been established that not all knowledge is beneficial for downstream tasks [68]. As depicted in the third example of Figure 1, an indiscriminate knowledge fusion as a prompt may generate harmful contexts or fabricated facts (e.g., regard Marie Curie as a physicist), which deteriorate the knowledge transfer from pre-training tasks to downstream tasks (a.k.a., the issue of "negative transfer" [43]). Additionally, unlike human beings, LLMs and GNNs are both vulnerable to noises and perturbations on the contextual prompts, which do not change the meaning of original tasks. For example, Figure 2 shows that even with minor changes in a graph or textual prompt, the state-of-the-art methods reveal dramatic performance variations for dedicated tasks within a single domain. Such vulnerability would be exacerbated in cross-domain tasks since the prompt is more prone to carry misinterpreted negative knowledge from various modalities.

To overcome the aforementioned challenges, we propose a novel cross-domain prompting framework for both language and graph tasks. The key insight of this framework involves jointly training two reinforced agents to select a subset of graph edges as external knowledge for improving the multi-domain model capabilities. Our framework can be easily implemented in two popular learning modes, i.e., "supervised learning" [23], and "pre-train and fine-tune" [32]. Specifically, we first introduce a highly scalable prompt to enrich and share knowledge between textual and graph data. Following the idea of thought prompting (e.g., CoT [61]), we decompose the textual input into multi-step reasoning questions, which are semantically and logically mapped to relevant graph nodes. By linking relevant nodes together as edges, we create a graph-like prompt coupled with associated questions to transfer knowledge between modalities. Then, we propose a reinforcement learning-based method to learn the optimal edge selection strategy for multi-domain knowledge extraction. In particular, we model the edge selection problem as a combinatorial optimization problem and propose a joint learning framework to optimize the edge selection strategy that maximizes the inner rewards of two policy-based agents: the L-agent dedicated to the language tasks, and the G-agent targeted at the graph tasks. During model training, we develop a joint multi-view optimization module to mitigate the issue of negative transfer through agent-level collaboration, where we create a hybrid reward function to provide a credible and diverse evaluation of selected edges.

Our main contributions are summarized as follows: (i) To the best of our knowledge, we are among the first to propose a cross-domain prompting approach that mutually aids diverse tasks of language and graph domains; (ii) We propose a domain-scalable prompt to capture the semantic and logical consistency from multi-modal data; (iii) We propose a lightweight plug-and-play framework that assigns two reinforced agents to select optimal edges for task-useful knowledge to enhance the model learning; (iv) Extensive experiments conducted on real-world language and graph datasets demonstrate the effectiveness, robustness, and interpretability of our framework in promoting state-of-the-art graph and language models under different learning modes.

2 PRELIMINARIES

In this section, we briefly introduce three essential concepts related to our work, namely language models, graph models, and prompt learning respectively.

Language Models. Language models have brought unprecedented performance in various Natural Language Processing (NLP) applications, including sentiment analysis [58], question answering [5], and text summarization [28]. One key contributor is the introduction of the "pre-train and fine-tunes" paradigm that first pre-trains the models on diverse, large-scale corpora for prior knowledge and subsequently fine-tunes them for downstream tasks [75]. In general, the fine-tuning objective of a NLP task can be formulated as:

$$\min_{\phi_l} \mathcal{L}_l[\mathcal{F}_{\phi_l}(X_l), Y_l], \tag{1}$$

where $\mathcal{F}_{\phi_l}(\cdot)$ is a pre-trained language model parameterized by ϕ_l. X_l and Y_l denote the input features and ground-truth labels of a language task, respectively.

Graph Models. For graph-based applications (e.g., link prediction [55]), graph models have risen as a powerful tool that offers the impressive ability to decode and interpret complex graph data [6, 17, 39]. The core of graph models lies in their ability to employ either embedding-based methods (e.g., TransE [2], ANALOGY [26]) or message-passing mechanisms (e.g., GAT [51], GraphSAGE [13]) as a pivotal component for learning the effective representations of graph structural information. Besides conventional supervised learning, the prevalence of "pre-train and fine-tune" paradigm has sparked research in developing various graph pre-training strategies to learn transferable knowledge for downstream tasks [31, 74]. Typically, these methods pre-train a graph model with easily accessible data, and then transfer the graph knowledge to a new domain or task by tuning the last layer of the pre-trained model. Here, the general objective of a graph learning task can be formulated as follows:

$$\min_{\phi_g} \mathcal{L}_g[\mathcal{F}_{\phi_g}(\mathcal{X}_g, \mathcal{A}), Y_g], \tag{2}$$

in which $\mathcal{F}_{\phi_g}(\cdot)$ represents a randomly-initiated or pre-trained graph model parameterized by ϕ_g. $\mathcal{G}$, X_g, Y_g, $\mathcal{A}$ represent the input graph, node features, ground-truth labels, and adjacency matrix of a graph task, respectively.

Prompt Learning. Prompt learning has been widely used in language and graph domains to improve performance [31, 33, 45, 74]. For instance, in the language domain, the research by OpenAI [33] enhanced Large Language Models (LLMs) with readable instructions by adding a textual prompt to a certain position of the input text. In the graph domain, Sun et al. [45] modified the original graph with an inserted graph prompt to improve the downstream performance of a pre-trained Graph Neural Network (GNN). More recently, the concept of prompt learning has been successfully introduced to exploit multimodal knowledge for dedicated tasks. For instance, various textual instructions were designed to study the correlation between topological knowledge and textual semantics for graph learning [69]. Indeed, these prompt designs liberate conventional knowledge fusion methods from relying on dense neural modules to explore rich multimodal information [12, 66], and act as potential tools to bridge the gap across different modalities. However, they are still in the early stages and lack a detailed examination of the complexity and interpretability involved in cross-modal interactions. Moreover, their explorations have been limited to the "pre-train and fine-tune" setting and overlook the realm of "training from scratch", which is a proven effective training technique in diverse applications, including text summarization [24, 29], link prediction [68], and object recognition [77].

Problem Definition: The objective of this research is to develop a lightweight and effective cross-domain prompting function, denoted as $F_{\phi_p}(\cdot)$, that is capable of relieving the difficulties of transferring prior knowledge to different domains, and facilitating the mutual learning of language model $\mathcal{F}_{\phi_l}(\cdot)$ and graph model $\mathcal{F}_{\phi_g}(\cdot)$ across their corresponding tasks.

3 METHODOLOGY

In this section, we present an in-depth analysis of our cross-domain prompting framework for graph and language tasks. Section 3.2 delves into the cross-domain prompting problem, which centers

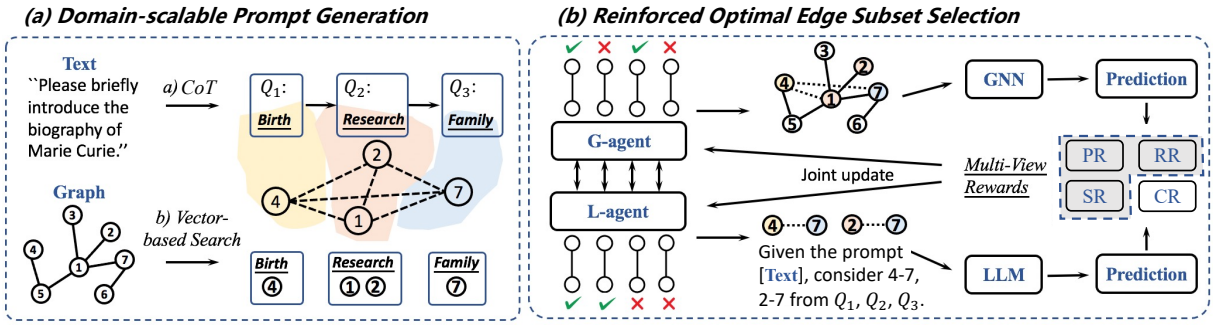


Figure 3: The overview of Cross-Modal Reinforced Prompting (CMRP) Framework.

on the dynamic selection of a subset of edges to enhance tasks in different modalities. Section 3.3 introduces our domain-scalable prompt that can be easily tailored to different modalities. In Section 3.4, we mathematically model the selection of the optimal subset of knowledge edges as a combinatorial optimization problem and develop a collaborative policy learning scheme, which mutually optimizes dual-domain agents for multiple dedicated tasks.

3.1 Framework Overview

Figure 3 illustrates the framework of **C**ross-**M**odal **R**einforced **P**rompting (**CMRP**), which targets at jointly enhancing the performance of graph models (e.g., GNNs) and language models (e.g., LLMs) in their domain-specific tasks. Our method proposes a new prompting mechanism, namely dynamic edge selection, to augment models with useful knowledge tailored to their specific tasks. Specifically, given a pair of text-graph inputs, we first convert the textual input into a domain-scalable prompt, which not only retains the semantic and logical contents of the original input but also highlights related graph information as external knowledge for different domains. Then, we propose a collaborative policy learning scheme to optimize the edge selection strategy that maximizes the mutual rewards. In particular, we design two agents, L-agent and G-agent, to perform tasks in different domains and search for the optimal subset. L-agent enhances the task performance of LLMs with subgraph-based textual prompts, and G-agent enriches graph tasks with new edges for GNNs. Later, a joint multi-view optimization module is developed to encourage two agents to share training experiences and identify the negative knowledge transfer through collaborative and contrastive rewards.

3.2 Cross-modal Prompting Formulation

Beyond building various exquisite deep networks for knowledge fusion, our key insight is to actively distill useful knowledge from multimodal data and seamlessly integrate it for dedicated tasks. However, the definition of knowledge usefulness is ambiguous and varies from domain to domain. For example, in the graph domain, useful textual knowledge may complement missing yet critical edges inside the graph, helping graph models exploit hidden topological information to perform better at all node-level, edge-level, and graph-level tasks. In the language domain, the useful graph knowledge may correspond to the task-relevant edges, which can be reconfigured into textual prompts to augment original textual input. Notably, graphs are the main modality of data we receive from nature [50], and most of the patterns we see are elegantly representable using the graph structures. Therefore, without loss of generalization, we use graph structure as an effective channel to transfer knowledge across different modalities.

Specifically, we first collect knowledge relevant to the downstream tasks in different modalities and then convert them into a graph format. Subsequently, we develop a dynamic edge selection approach to selectively extract a subset of edges from the graph. These edges are utilized as graph prompts, tailored to boost the performance of the specific task. Formally, based on the Eq. (1) - (2), given a pair of graph and text inputs, the cross-modal prompting problem can be formulated as:

$$\min_{\phi_l,\phi_g} \mathcal{L}\left[\mathcal{F}_{\phi_l}\left(\mathcal{F}_p(\mathcal{X}_l, P_l)\right), Y_l\right] + \mathcal{L}\left[\mathcal{F}_{\phi_g}\left(\mathcal{X}_g, \mathcal{F}_p(\mathcal{A}, P_g)\right), Y_g\right], \quad (3)$$

in which our prompting function $\mathcal{F}_p(\cdot, P) = \pi_\star(S_\star(\cdot, P_\star))$[1] aims to inject the domain-scalable prompt $P_\star$ into the original inputs based on the inserting function $S_\star$ (Section 3.3), and then utilize the reinforced agent with policy $\pi_\star$ to extract edges with optimal cross-domain knowledge (Section 3.4).

3.3 Domain-scalable Prompt Design

Unifying Prompt for Multimodal Data. Developing a unified prompting mechanism is crucial for interacting with multimodal data; otherwise, it leads to increased algorithm complexity for cross-modal knowledge transfer [43]. To achieve this, one challenging aspect is to bridge the knowledge gap between language and graph data effectively. Following [45], we first analyze the knowledge characteristics of language and graph domain and discover that a fair domain-scalable prompt should contain at least two components: *(1) prompt structure*, which indicates the connection of different tokens from logical and semantic aspects. Generally, the language prompt tokens (e.g., words [27]) are preset as a linear relation like a sub-sentence or a phrase, while the graph prompt tokens (e.g., dummy tokens [45]) are non-linearly connected, thus making the logic of graph prompt far more complicated to comprehend than language prompt; *(2) injection mechanism*, which denotes how to inject the prompt into the input data. The language prompt can be directly added to the default positions like the rear of the input

[1]The subscript ⋆ stands for a particular domain, either g (i.e., graph domain) or l (i.e., language domain).

texts, whereas the graph prompt is more difficult to inject, as it has no explicit positions like a sentence to pair with the input graph.

3.3.1 Prompt Structure. Unlike the continuous word representations of textual knowledge, graph nodes are discretely connected and convey more abstract knowledge. To narrow their knowledge gap, as shown on the left side of Figure 3, we first introduce the thought prompting method (e.g., Chain-of-Thought (CoT) [61]) for hierarchical logical reasoning, which guides LLM to decompose the original input text (e.g., "Please briefly introduce the biography of Marie Curie") into intermediate reasoning question set $\mathcal{Q} = \{Q_1, Q_2, \dots\}$. We notice that each question focuses on a distinct topic, e.g., "Birth", "Research", or "Family", and usually requires one-word answers, which allow us to map these questions (e.g., Q_1) to relevant graph nodes (e.g., ④) via the vector-based search [10, 11, 14]. Motivated by this, we transform the textual input into a new prompt structure $\mathcal{P} = (\Omega, P)$, where $\Omega = \{\omega^1, \dots, \omega^{|\Omega|}\}$ denotes the set of prompt tokens and $|\Omega|$ is the number of prompt tokens. In practice, we usually have $|\Omega| \ll |V|$, in which $|V|$ denotes the number of input graph nodes. Also, the node-wise correlations between and within questions are equally important to investigate. For example, from the research aspect, the discovery of radium (e.g., ①) made Marie Curie win the Nobel Price (②), which was also co-won by her husband Pierre Curie (⑦). Thus, we connect all these tokens as an edge set $P = \{p^1, \dots, p^T\}$, where T is the number of total edges. Based on the above discussions, such prompt structure reveals two major benefits: (1) For the graph task, we can exploit the node-wise relations from a textual perspective and supplement the missing yet important graph links (e.g., ② − ⑦) to enhance the training of graph models. (2) In terms of the language task, we can regard these questions and prompt tokens as a sequence of thoughts and answers, which can provide user-friendly explanations for downstream tasks.

3.3.2 Injection Mechanism. To better illustrate, we use $P_g \subseteq P$, $P_l \subseteq P$ to represent the prompt edges injected in the graph and language domains, respectively. For the graph task, the injection strategy S_g involves a direct aggregation of the prompt and graph edges P_g, $\mathcal{E}$ because of their identical modality. The merged edges are subsequently used to train graph models for all graph-level tasks without any modifications to graph models. For the language task, the injection strategy S_l is to first extract the M-hop subgraphs of prompt tokens of P_l, which are then converted into the textual input associated with its corresponding questions. Finally, these prompts are used to instruction-tune language models for various downstream tasks in a few-shot or zero-shot manner. A practical illustration of this process can be found in the lower middle of Figure 3 (b). However, an indiscriminate knowledge fusion may result in deteriorated model performance [18]. This is due to that not all knowledge is useful for downstream applications [34], and considering all edges for multi-domain tasks would incur considerable computational overhead.

3.4 Reinforced Optimal Edge Subset Learning

In this section, we formulate the problem of selecting the optimal subset of prompt edges as a combinatorial optimization problem. Given a specific task $\mu_\star$, the objective is to learn a stochastic policy $\pi_{\theta_\star}(P_\star^t \mid \mu_\star)$ using the chain rule to factorize the probability of selecting a subset $P_\star^t$ of edges from $P_\star$. The policy network uses this information to determine the optimal subset of nodes to select to explore the most useful task-related knowledge at each training iteration t as follows:

$$\pi_{\theta_\star}(P_\star^t \mid \mu_\star) = \prod_{t=1}^{T} \pi_{\theta_\star}(p_\star^t | \mu_\star, p_\star^{1:t-1}). \tag{4}$$

3.4.1 Policy Network Design. As shown in Figure 3 (b), to perform the tasks in graph and language modalities, we design two policy-based agents, namely G-agent and L-agent, to make optimal edge selection respectively. In particular, G-agent and L-agent adopt two separate yet identical policy networks π_{θ_g} and π_{θ_l}, which take the edge features as input and construct the edge pool $P_\star^t$ in an auto-regressive manner for downstream tasks. According to Eq. (4), each agent selects one edge at a time and uses the previous selections to decide the next edge until every edge is enumerated. Note that a given prompt edge $p_t \in P$ usually consists of the head entity and tail entity. To alleviate the computational burden, we skip the process of searching for their potential relation from graph and construct a relation-free prompt edge as $p_t = \{e_t^{head}, e_t^{tail}\}$.

Here, we use the corresponding model (e.g., GNN) to encode p_t into three d_i-dimensional embedding vectors and linearly project these concatenated vectors from $\mathbb{R}^{2d_i}$ to $\mathbb{R}^{d_i}$.

Inspired by [73], we further use these embeddings to represent RL actions and states of two agents. The actions of both G-agent and L-agent correspond to their individual selection or deselection of current prompt edge p_t. Meanwhile, for π_{θ_g} and π_{θ_l}, we incorporate two separate LSTMs [16] to encode their selection history $h_\star^t \subseteq \{p_\star^1, \dots, p_\star^t\}$ according to the recurrent dynamics,

$$\mathbf{h}_g^1 = \text{LSTM}_g(\mathbf{0}, \mathbf{p}_1),\ \mathbf{h}_l^1 = \text{LSTM}_l(\mathbf{0}, \mathbf{p}_1), \tag{5}$$

$$\mathbf{h}_g^t = \text{LSTM}_g(\mathbf{W}_g \mathbf{h}_g^{t-1},\ \mathbf{p}_g^t),\ t > 1, \tag{6}$$

$$\mathbf{h}_l^t = \text{LSTM}_l(\mathbf{W}_l \mathbf{h}_l^{t-1},\ \mathbf{p}_l^t),\ t > 1, \tag{7}$$

where $\mathbf{W}_g, \mathbf{W}_l \in \mathbb{R}^{2d_i \times 2d_h}$ are the tunable weight matrices.

To predict the next action of G-agent and L-agent, we further utilize a two-layer feedforward network to process the concatenation of their last LSTM states and current RL state embeddings,

$$\pi_{\theta_g}(a_g^t | P_g^t) = \sigma(\mathbf{W}_g^2\ \text{ReLU}(\mathbf{W}_g^1[\mathbf{p}_g^t; \mathbf{h}_g^t])), \tag{8}$$

$$\pi_{\theta_l}(a_l^t | P_l^t) = \sigma(\mathbf{W}_l^2\ \text{ReLU}(\mathbf{W}_l^1[\mathbf{p}_l^t; \mathbf{h}_l^t])), \tag{9}$$

where $\mathbf{W}_g^1, \mathbf{W}_g^2, \mathbf{W}_l^1, \mathbf{W}_l^2 \in \mathbb{R}^{4d_h \times 4d_h}$ are the matrices of learnable weights, and σ denotes the softmax operator. a_g^t, a_l^t represent the binary decisions made by G-agent and L-agent, which determine the selection of the current edges p_g^t, p_l^t.

3.4.2 Joint Multi-view Optimization. If the default rewards for G-agent and L-agent only consider how well their corresponding graph and language models perform in domain-specific tasks, the dual agents would encounter two serious problems that hinder their knowledge transfer abilities. First, as the graph contains more abstract information than texts, we need to carefully examine the knowledge consistency when fusing and applying them to domain-specific tasks. Second, G-agent and L-agent do not share any cross-domain task experience to improve their performances. To alleviate

the issue, we provide hybrid reward signals to evaluate the effectiveness of an agent's edge selection from both individual and contrastive perspectives:

(i) Task Rewards. Given a task $\mu_\star$, we design three different rewards, i.e., *Performance Reward (PR)*, *Reasoning Reward (RR)*, and *Semantic Reward (SR)*. In particular, the PR is first proposed to measure the effectiveness of a model based on its performance score$_\star(\cdot)$ on the test set:

$$\mathcal{R}_\star^{PR} = \text{score}_\star(\mu_\star). \tag{10}$$

Then, we utilize the RR to encourage an agent to respect the coherence of intermediate reasoning questions, where we compute the proportion of unique questions that are covered by selected edges:

$$\mathcal{R}_\star^{RR}(t) = \mathbb{E}_{Q_i \in \mathcal{Q}}\left[\sum_{\substack{p_\star^t \in P_\star^t \\ \omega_\star^j \in p_\star^t}} \mathbb{1}\left[\exists \psi(\omega_\star^j) = Q_i\right]\right], \tag{11}$$

where $\psi(\cdot)$ is a mapping function that links each prompt token with its corresponding question. After that, the SR is developed to help the agents maintain a high degree of semantic similarity between selected edges and the original prompt edges:

$$\mathcal{R}_\star^{SR}(t) = \mathbb{E}_{p_\star^t \in P_\star^t}\left[\frac{\sum_{p \in P} \Phi(\mathbf{p}, \mathbf{p}_\star^t)}{T}\right], \tag{12}$$

in which $\Phi(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^\top \mathbf{y}}{||\mathbf{x}||_2 ||\mathbf{y}||_2}$ is calculated as the cosine similarity between two embeddings.

(ii) Contrastive Rewards. In order to prevent negative transfer between modalities [70, 78], we further construct a *Contrastive Reward* (CR), where we assign larger rewards to the positively-related input pairs $\mathcal{I}^+$ to speed up model convergence, and lower rewards to the negatively-related ones $\mathcal{I}^-$ to reduce noisy knowledge:

$$\mathcal{R}_\star^{CR} = -\mathop{\mathbb{E}}_{j \in \mathcal{I}^-} \mathcal{R}_\star^j + \mathop{\mathbb{E}}_{i \in \mathcal{I}^+} \mathcal{R}_\star^i, \tag{19}$$

in which $\mathcal{R}_\star = \mathcal{R}_\star^{PR} + \sum_{t=1}^{T}\left(\sigma \mathcal{R}_\star^{RR}(t) + \beta \mathcal{R}_\star^{SR}(t)\right)$, where σ, β are hyperparameters that control the importance of individual rewards.

Model Training. The detailed training procedure of CMRP is described in Algorithm 1. Line 3 generates the domain-scalable prompt $\mathcal{P}$. During edge subset construction, we use lines 4-8 to iteratively calculate the conditional probability of each edge to be selected. Line 9 computes the contrastive aggregated rewards for two agents. Line 10 inserts the prompt subsets into two domain-specific tasks by modifying the original training inputs. In line 11, we leverage new training inputs to regularize the training of graph and language models. Line 12 applies the REINFORCE [73] to optimize the edge selection ability of dual agents, with a combined moving average baseline to reduce variance. Line 13 updates the baseline for each agent based on their weighted mutual rewards.

3.4.3 Computational Complexity. Recall that an input graph has $|V|$ nodes, the feature sizes of input data and hidden layer are denoted by d_i and d_h, respectively. The time complexity of our reinforcement learning framework is $O(Td_h^2 + Td_hd_i)$. If we apply the "supervised learning" or the "pre-train and fine-tune" to update the graph model, the space and time complexity of graph models (e.g., L-layer GCN) would remain almost unchanged to be $O(|V|^2 + Ld_h^2 + L|V|d_h)$ and $O(L|V|^2d_h + L|V|d_h^2)$, respectively. This is because our method is designed to augment the original graph connections with prompt edges without changing the model structure. Similar conclusions can be applied to the pre-trained language models, which are fed with reinforced prompts as in-context knowledge under the zero-shot or few-shot settings.

Algorithm 1 CMRP Training Algorithm

1: **Input:** Given a pre-trained or randomly-initiated graph model $\mathcal{F}_{\phi_g}$ and a pre-trained language model $\mathcal{F}_{\phi_l}$; A pair of graph task μ_g and language task μ_l; Adjacency matrix $\mathcal{A}$ of input graph $\mathcal{G}$ and node features $\mathbf{X}_g$; Input textual features $\mathbf{X}_l$; Groundtruth labels of graph and language tasks $\mathbf{Y}_g$, $\mathbf{Y}_l$; Episode size E

2: **Output:**Well-trained π_{θ_g}, π_{θ_l}, $\mathcal{F}_{\phi_g}$, and $\mathcal{F}_{\phi_l}$

3: Create a domain-scalable prompt $\mathcal{P}$ through the CoT reasoning and conduct the vector-based search

4: **for** episode e in $\{1, \ldots, E\}$ **do**

5: Initialize the edge subsets P_g^0, P_l^0 of G-agent , L-agent as empty sets

6: **for** $t = 1, \ldots, T$ **do**

7: Predict the possibilities of adding current edges p_g^t, p_l^t into P_g^t, P_l^t based on Eq. (8) - (9)

8: **end for**

9: Compute $\mathcal{R}_g^{CR}$, $\mathcal{R}_l^{CR}$ based on Eq. (19)

10: Inject P_g^T, P_l^T into graph and language domains, and modify the original inputs based on Eqn. (3)

11: Perform the model training of graph and language models:

$$\phi_g \leftarrow \phi_g - \eta_g \cdot \nabla_{\phi_g} \mathcal{L}(\mathcal{F}_{\phi_g}(X_g, \mathcal{A}'), Y_g) \tag{13}$$

$$\phi_l \leftarrow \phi_l - \eta_l \cdot \nabla_{\phi_l} \mathcal{L}(\mathcal{F}_{\phi_l}(X_l'), Y_l) \tag{14}$$

12: Use the REINFORCE to update the parameters of dual agents:

$$\theta_g \leftarrow \theta_g + \eta_p \cdot \nabla_{\theta_g}(\mathcal{R}_g^{CR} - B_g^t) \tag{15}$$

$$\theta_l \leftarrow \theta_l + \eta_p \cdot \nabla_{\theta_l}(\mathcal{R}_l^{CR} - B_l^t) \tag{16}$$

13: Update the Baselines B_g^t, B_l^t for dual agents:

$$B_g^{t+1} \leftarrow \gamma_1 B_g^t + (1-\gamma_1)(\mathcal{R}_g^{CR} + \lambda_1 \mathcal{R}_l^{CR}) \tag{17}$$

$$B_l^{t+1} \leftarrow \gamma_2 B_l^t + (1-\gamma_2)(\mathcal{R}_l^{CR} + \lambda_2 \mathcal{R}_g^{CR}) \tag{18}$$

14: **end for**

15: **return** $\pi_{\theta_g}, \pi_{\theta_l}, \mathcal{F}_{\phi_g}, \mathcal{F}_{\phi_l}$

4 EXPERIMENTS

In this section, we compare the performance of our proposed method to the state-of-the-art (SOTA) graph models, language models, and prompting methods on various graph and language tasks. We aim to answer the following research questions: **RQ1**: How effectively does our method improve the state-of-the-art graph and language methods under the "supervised learning" or "pre-train and fine-tune" paradigm? **RQ2**: How do the main components of our method impact the performance? **RQ3**: How robust is our method with respect to noisy or complex inputs? **RQ4**: How powerful is our method in enhancing the interpretability of graph and language models? Due to space limit, we put the detailed descriptions of datasets, model implementations, and additional experiments in Appendix A.1-A.3, respectively. The source code and data are publicly available at https://github.com/JohnJiang12138/CMRP

4.1 Experimental Setup

Datasets. We conduct an extensive comparison of our methods with other approaches on **23** public and synthetic datasets, including **(i) thirteen graph learning datasets:** Node Classification (NC), Link Prediction (LP), Graph Classification (GC); and **(ii) ten natural language understanding datasets:** Publication Classification (PC), Question Answering (QA), Movie Classification (MC). Throughout model training and inference, we consider the inherent correlation between graph and language tasks, thus forming cross-domain input pairs as (NC, PC), (LP, QA), and (TS, MC).

Baselines. To verify the effectiveness of our CMRP, we select 19 SOTA baselines, which include:

(i) Six graph embedding based methods:

- TransE [2] follows the geometric principle as a translation to encode the relationships among all entities and relations into the low-dimensional embeddings.
- TransH [59] is a variant of TransE, which models a relation as a hyperplane together with a translation operation on it.
- TransD [20] is another variant of TransE, which provides a flexible mapping matrix to project entity embeddings to relation subspace.
- ANALOGY [26] models analogical structures in multi-relational embedding via a differentiable loss function.
- SimplE [21] develops the non-negativity on entity representations and approximate entailment constraints on all relation representations.
- ComplEx [49] proposes a simple approach to matrix and tensor factorization for link prediction task.

(ii) Six GNN-family models:

- GCN [22] is a semi-supervised convolutional network that operates directly on graphs.
- GAT [51] leverages masked self-attentional layers to attend over node neighborhoods' features for graph convolutions.
- GT [72] proposes a generalized transformer neural architecture to learn topological information.
- RESGCN [41] introduces three key components, i.e., the residual links, bottleneck structure, and part-wise attention to improve GCN.
- JKNET [19] introduces a dense convolutional network to feed-forwardly connect each layer to every other layer.
- INCEPGCN [46] designs suitably factorized convolutions and aggressive regularization to simply the architecture of deep convolutional networks.

(iii) Three LLM-family models:

- ChatGPT-3.5 Turbo [33] is a decoder-only foundation model, which is pre-trained and fine-tuned to follow instructions in a prompt and provide a detailed response.
- LLama2 7b Chat [48] is another pre-trained foundation model meant to be fine-tuned for specific use cases.
- ChatGLM-3 Turbo [7] is a general pre-trained language model based on auto-regressive blank infilling for various downstream tasks.

(iv) Four prompting methods:

- ProG [45] develops a graph prompt to fine-tune the pre-trained GNNs for graph learning tasks.
- InstructGLM [69] proposes some natural language prompts to instruction-tune LLMs for graph learning tasks.
- Dropedge [37] is an augmentation-based method that provides a pseudo edge-dropping prompt by randomly removing a certain amount of edges from the input graph to reduce graph message passing.
- G-mixup [15] is another augmentation-based "prompting" technique, which interpolates graphons of different classes in the Euclidean space to get mixed graphons.

Evaluation Metrics. Our evaluation employs different metrics tailored to each specific task. Specifically, for LP task, we use the filtered ranking-based metrics [56], i.e., Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hits@k (k=1, 3, 10). In terms of NC and GC tasks, we apply the classification accuracy metric (%) to evaluate the model's performance [15]. When it comes to PC, QA, and MC tasks, we re-configure them into a unified format of QA template, and follow [42] to adopt the exact match accuracy (Hits@1) as the evaluation metric.

4.2 RQ1: Overall Performance

To answer the RQ1, we assess the effectiveness of our method CMRP in boosting the performance of the SOTA graph and language models under different learning paradigms: **(i) "Supervised learning" on graph baselines.** We compared our prompt-based methods with other prompt-free baselines on edge-level, node-level, and graph-level tasks. We repeat the evaluation 5 times and report the average results in Table 1, Table 2, and Table 3. Compared to all baselines, the average percent improvements are (11.91%, 12.40%, 11.61%) on three NC datasets, (2.11%, 4.41%, 1.3%, 2.62%,10.31%) on five LP datasets, (4.77%, 10.64%, 10.8%, 12.86%, 3.01%) on five GC datasets given removing outlier data. The experimental results show that our CMRP can yield more comprehensive and stable improvements in the performance of various graph models. **(ii) "Pre-train and fine-tune" on GNNs.** We further examine the elevated performances of the pre-trained GNNs using our CMRP compared to the SOTA ProG, Dropedge, and G-mixup. In particular, we first use SimGRACE [64] to pre-train a GNN model in a self-supervised manner, and then utilize a prompt to optimize the fine-tuning of the pre-trained model for a downstream task. The results reported in Table 2 demonstrate that our method is more capable of distilling useful knowledge from multi-domain data, exemplified by a 11.91%, 12.40%, and 11.61% improvement on Cora, CiteSeer, and Pubmed. **(iii) "Pre-train and fine-tune" on LLMs.** Table 1 depicts the overall performance of our CMRP and one prompting baseline InstructGLM on three language tasks under the zero-shot or few-shot setting. Compared to the baselines, CMRP receives the average performance gains of 6.9% on PC task, 4.2% on QA task, 11.9% on MC task by removing outlier data. The results again verify the effectiveness of CMRP in exploiting useful topological information as in-context knowledge to enhance the performance of LLMs in various language tasks. Notably, InstructGLM displays high standard deviations compared to CMRP on these NC datasets, which indicate the graph and language prompts have a relatively low signal-to-noise ratio, making the pre-trained GNNs easier to get distracted by these prompts.

Table 1: Performance comparison of our model and the SOTA baselines on LP and QA tasks. For simplicity, we omit the standard deviations which vary in a small range. "ZS" and "FS" denotes "Zero-Shot" and "Few-shot" respectively.

Graph	WN18				FB15K				WN18RR				FB15K237				YAGO3-10			
Dataset	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
TransE	50.5	9.0	92.7	94.7	52.3	37.5	63.1	76.2	19.3	2.3	37.0	44.7	28.6	19.0	32.5	47.5	31.0	19.9	36.6	52.4
TransE+Ours	53.3	13.9	**92.9**	**94.8**	54.0	39.4	64.7	77.4	19.6	2.3	37.6	45.1	**28.8**	**19.2**	32.6	47.8	35.3	23.7	41.6	56.9
TransH	47.7	5.0	91.3	94.0	48.7	32.0	61.3	74.9	18.5	1.5	35.7	41.8	28.2	17.9	32.6	48.0	41.0	29.8	47.6	61.3
TransH+Ours	49.1	6.6	92.1	94.1	51.6	35.2	64.3	77.3	18.8	1.2	36.8	42.5	**28.8**	18.6	**33.1**	**48.9**	**43.6**	**32.7**	**50.1**	**63.3**
TransD	49.2	7.6	91.6	94.1	54.2	39.5	65.0	77.6	18.6	2.5	36.0	42.8	28.1	17.7	32.5	48.2	37.1	25.8	43.2	58.1
TransD+Ours	51.5	11.1	92.2	94.1	**56.4**	**42.0**	**68.1**	**79.5**	19.0	2.8	36.8	42.9	28.7	18.3	33.0	48.8	41.9	31.2	48.1	62.2
ANALOGY	72.1	57.1	86.2	92.3	46.2	32.3	54.7	71.7	40.1	36.1	42.4	46.9	25.4	16.6	28.5	42.8	18.4	5.3	24.4	44.5
ANALOGY+Ours	72.7	57.5	87.2	92.6	47.9	33.7	56.8	73.6	**40.7**	**36.7**	**42.6**	**47.9**	25.7	16.9	28.9	43.5	25.2	15.1	29.2	45.7
SimplE	46.6	32.3	54.0	76.4	22.7	12.7	25.0	44.1	32.7	25.0	38.3	46.0	18.4	9.9	21.2	35.5	7.5	2.9	6.8	17.4
SimplE+Ours	48.5	34.5	56.0	76.6	24.5	14.2	27.5	45.1	34.2	27.2	38.7	46.2	19.5	11.6	21.4	36.6	9.9	5.2	9.7	18.8
ComplEx	71.7	61.1	80.0	89.4	35.5	22.9	41.4	60.6	39.3	35.1	42.0	45.7	24.2	15.8	26.9	41.0	10.7	2.9	11.8	26.8
ComplEx+Ours	**73.8**	**64.1**	81.5	90.1	36.0	22.9	42.7	61.0	39.7	35.5	42.3	46.0	24.7	16.3	27.5	41.7	15.3	8.4	15.7	30.3
Language	SimpleQA		CWQ		SimpleQA		CWQ		SimpleQA		CWQ		SimpleQA		CWQ		SimpleQA		CWQ	
Dataset	ZS	FS	ZS	FS	ZS	FS	ZS	FS	ZS	FS	ZS	FS	ZS	FS	ZS	FS	ZS	FS	ZS	FS
ChatGPT-3.5 Turbo	0.57	0.49	0.47	0.82	0.49	0.35	0.68	0.70	**0.52**	0.49	0.72	0.61	0.76	0.62	0.65	0.54	0.33	0.41	0.68	0.66
ChatGPT-3.5 Turbo + Ours	**0.60**	**0.53**	0.50	**0.84**	**0.50**	0.34	**0.71**	0.69	0.50	**0.50**	**0.75**	**0.65**	**0.79**	0.63	**0.70**	**0.58**	**0.35**	0.42	0.69	**0.69**
LLama2 7b Chat	0.56	0.17	0.52	0.40	0.25	0.19	0.48	0.30	0.49	0.48	0.61	0.52	0.60	0.61	0.64	0.44	0.22	0.21	0.63	0.31
LLama2 7b Chat + Ours	0.59	0.20	0.56	0.46	0.28	0.19	0.46	0.33	0.51	**0.50**	0.63	0.55	0.61	0.62	0.66	0.48	0.20	0.20	0.67	0.33
ChatGLM-3 Turbo	0.52	0.35	0.76	0.65	0.43	0.37	0.68	0.72	0.48	0.48	0.63	0.58	0.70	0.62	0.48	0.24	0.22	**0.44**	**0.73**	0.64
ChatGLM-3 Turbo + Ours	0.57	0.37	**0.80**	0.69	0.46	**0.38**	0.70	**0.75**	0.51	0.46	0.63	0.59	0.71	**0.64**	0.51	0.27	0.26	0.43	**0.73**	0.65

Table 2: The experimental results of our model and the closest baseline ProG [45] and InstructGLM [69] on both NC and PC tasks, respectively. The best performance on each dataset is highlighted in bold.

Graph Dataset	Cora	CiteSeer	PubMed
SimGRACE+GAT	71.60±1.02	74.35±0.18	69.93±0.47
SimGRACE+GAT+ours	78.49±0.40	82.40±0.39	75.20±0.95
SimGRACE+GCN	58.49±0.47	69.57±0.57	52.40±0.61
SimGRACE+GCN+ours	78.46±0.92	**82.80**±0.61	81.33±0.36
SimGRACE+GT	62.38±1.67	57.98±3.47	51.27±0.95
SimGRACE+GT+ours	77.68±0.33	82.04±0.37	76.45±1.87
RESGCN	84.56±0.445	76.76±0.26	89.10±0.29
RESGCN+ours	**85.06**±0.29	77.96±1.5	89.74±0.29
JKNET	83.70±1.05	77.56±0.32	89.44±0.35
JKNET+ours	84.76±0.22	78.40±0.40	89.86±0.68
INCEPGCN	83.10±0.93	77.76±0.50	89.76±0.41
INCEPGCN+ours	84.21±0.58	78.08±0.19	**89.94**±0.59
Language Dataset	PC-Cora	PC-CS	PC-PM
LLama2 7b Chat	0.76±0.16	0.26±0.12	0.10±0.17
LLama2 7b Chat + Ours	0.81±0.11	0.31±0.13	0.14±0.16
ChatGPT-3.5 Turbo	0.84±0.04	0.87±0.05	0.75±0.13
ChatGPT-3.5 Turbo + Ours	**0.87**±0.10	**0.88**±0.05	**0.78**±0.17

Table 3: Performance comparison of our model and the SOTA baselines on GC and MC tasks, with the best results in bold. Note that the MC task is performed in the context of zero-shot learning.

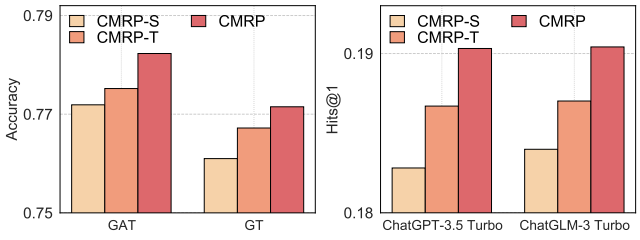
	Graph Dataset	IMDB-B	IMDB-M	REDD-B	REDD-M5	REDD-M12
GCN	vanilla	72.18±1.55	48.79±2.72	78.82±1.33	45.07±1.70	46.90±0.73
	Dropedge	72.50±0.31	49.08±1.89	81.25±8.15	51.35±1.54	47.08±0.55
	DropNode	72.00±4.09	48.58±2.85	79.25±0.35	49.35±1.80	47.93±0.64
	Subgraph	68.50±4.76	49.58±2.61	74.33±2.88	48.70±1.63	47.49±0.93
	M-Mixup	72.83±1.75	49.50±1.97	75.75±4.53	49.82±0.85	46.92±1.05
	G-Mixup	72.87±3.85	51.30±2.14	89.81±0.74	51.51±1.70	48.06±0.53
	Ours	**74.34**±1.12	**55.13**±2.01	**94.33**±0.55	**58.23**±0.30	**50.15**±0.34
GIN	vanilla	71.55±3.53	48.83±2.75	92.59±0.86	55.19±1.02	50.23±0.83
	Dropedge	72.20±1.82	48.83±3.02	92.00±1.13	55.10±0.44	49.77±0.76
	DropNode	72.16±0.28	48.33±0.98	90.25±0.98	53.26±4.99	49.95±1.70
	Subgraph	68.50±0.86	47.25±3.78	90.33±0.87	54.60±3.15	49.67±0.90
	M-Mixup	70.83±1.04	49.88±1.34	90.75±1.78	54.95±0.86	49.81±0.80
	G-Mixup	71.94±3.00	50.46±1.49	92.90±0.87	55.49±0.53	**50.50**±0.41
	Ours	**75.42**±1.57	**53.70**±1.97	**94.25**±0.66	**58.79**±0.74	50.09±0.56
	Language Dataset	MC-IB	MC-IM	MC-RB	MC-RM5	MC-RM12
LLM	LLama2 7b Chat	0.70±0.11	0.24±0.08	0.05±0.02	0.58±0.12	0.10±0.03
	LLama2 7b Chat + Ours	0.78±0.02	0.32±0.01	0.12±0.01	0.65±0.06	0.04±0.01
	ChatGPT-3.5 Turbo	0.57±0.05	0.68±0.07	0.70±0.02	0.82±0.04	0.48±0.14
	ChatGPT-3.5 Turbo + Ours	**0.92**±0.08	**0.92**±0.03	**0.80**±0.10	**0.90**±0.07	**0.67**±0.05

4.3 RQ2: Ablation Study

To answer RQ2, we examine the impact of key components of our reinforcement learning framework, i.e., different reward designs, on the accuracy of GAT and GT on the GC task, and the Hits@1 of ChatGPT-3.5 Turbo and ChatGLM-3 Turbo on the MC task. We investigate two variants of our method: (i) CMRP-S, which solely adopts the performance reward $\mathcal{R}_\star^{PR}$, (ii) CMRP-T, which abandons the contrastive rewards $\mathcal{R}_\star^{CR}$, and (iii) our method, which uses the CoT to guide prompt generation and incorporates multi-view rewards for multi-domain edge selection. As shown in Figure 4, the removal of any component leads to a notable decline in performance. In particular, through the comparison of CMRP-S and CMRP-T, we find it crucial to examine the knowledge consistency and semantic similarity of multi-domain inputs before fusing and applying them to domain-specific tasks. Lastly, it's worth noting that the performance will decrease 24.66% on average if we remove contrastive reward component for both tasks. This emphasizes that the hybrid reward plays a critical role in resolving the issue of negative transfer between modalities.

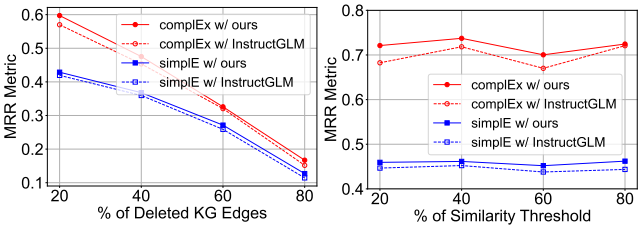
4.4 RQ3: Robustness Analysis

To answer RQ3, we perform a robustness analysis to evaluate the influence of noisy or complex inputs on the performance of our prompting framework using a LP dataset (WN18) and two QA datasets (SimpleQA [1], CWQ [47]) as an example. Specifically, to verify the robustness of our prompting method to noisy inputs, we either prune K% of the original graph edges or consider the top K% of semantically top-ranked graph nodes. As reported in Figure 5,



(a) NC task. (b) PC task.

Figure 4: Ablation study of multi-view rewards.



(a) Missing graph edges. (b) Varying semantic thresholds.

Figure 5: Robustness analysis of our CMRP under different conditions of noisy inputs.

compared to the InstructGLM, our method enables the underlying models to become relatively robust to these perturbations in a reasonable range from 0.167 to 0.597 in the first condition, and from 0.127 to 0.429 on QA task, indicating the good denoising ability of our CMRP. In addition, the hardness of questions is a key consideration in evaluating the reasoning ability of prompting approaches. We thus report the performance of ChatGPT-3.5 Turbo + Ours on complex reasoning QA dataset (CWQ) [47] and the simple QA dataset (SimpleQA) [1] in Table 1. The results demonstrate an overall stable trend in performance, with an average of 53.5% in the metric of accuracy, which shows our prompting method can well handle the deep reasoning problems.

4.5 RQ4: Case Study

To validate the ability of our reinforced prompt in enhancing model interpretability, we further study the intermediate reasoning questions provided by our prompts. Specifically, the prompt edges selected by dual agents are used to facilitate models to elicit human-understandable explanations for final inference. Table 4 shows two examples of LP and QA tasks using our CMRP based on a QA example and FB15K237 dataset. We can find that given the input text, the selected prompts closely adhere to the the reasoning questions outlined by CoT. For the LP task, we manually extract the links from the head entity "Courage Under Fire" to the tail entity "Detective Fiction", and the results highlight the logical consistency of our prompt-based path compared to the prompt-free one. Additionally, for the QA task, the LLM can generate answers reasonably grounded in the context provided by selected edges.

5 CONCLUSION

In this paper, we present a novel plug-and-play framework for incorporating a lightweight cross-domain prompting method into both language and graph learning tasks. To be more specific, we first proposed a domain-agnostic prompt to investigate the semantic and logical contents of the textual input but also highlight related graph information as external knowledge for different domain tasks. To distill useful knowledge from prompt edges, we propose a collaborative policy learning scheme to optimize the edge selection strategy that maximizes the mutual rewards of dual agents. We evaluate our approach on 23 real-world and synthetic graph and language datasets, and the results demonstrate its superiority over other baselines. Our method can effectively enhance the performance, robustness, interpretability of graph and language models under the "supervised learning" and "pre-train and fine-tune" paradigms.

Table 4: A case study of selected prompt edges to the model's reasoning ability under different tasks, where a selected edge is denoted as token1 (id_1) — token2 (id_2).

✿**Textual input:** "Name a detective film".
✿**Reasoning questions:** **Q1:** What language is spoken in the film? **Q2:** Where does the story of the film take place? **Q3:** Are there any other films that are similar in theme or style?
✿ **Selected prompt edges**: **E1:** "English Language" (160) — "United States of America" (32). **E2:** "The World Is Not Enough" (678) — "Family Plot" (1505). **E3:** "The World Is Not Enough" (678) — "United States of America" (32) **E4:** "Family Plot" (1505) —"Detective-GB" (9615)
✔ **Prompt for LP task**
Existing graph links: "The World Is Not Enough" $\xrightarrow{\text{film\_country}}$ "United States of America", "The World Is Not Enough" $\xrightarrow{\text{film\_release\_region}}$ "United States of America"
Test examples ("Courage Under Fire" $\xrightarrow{?}$ "Detective Fiction"):
Reasoning path w/o our prompt: "Courage Under Fire" $\xrightarrow{\text{film\_genre}}$ "Drama" $\xrightarrow{\text{tv\_program\_genre}}$ "Pushing Daisies" $\xrightarrow{\text{film\_genre}}$ "Detective Fiction"
Reasoning path w/ our prompt: "Courage Under Fire" $\xrightarrow{\text{film\_genre}}$ "Detective-GB" —"Family Plot" $\xrightarrow{\text{film\_genre}}$ "Detective Fiction"
✔ **Prompt for QA task**
Question: Please answer to the prompt "Name a detective film" based on the following information:
To answer the above question, we develop several intermediate reasoning questions **Q1**, **Q2**, **Q3**. According to these questions, the discovered graph knowledge include **E1**, **E2**, **E3**, **E4**.
Answer (ChatGPT 3.5): Based on the provided intermediate reasoning questions and graph knowledge, a suitable response to the prompt "Name a detective film" could be "Chinatown."
Reasons: (1) Language spoken in the film: English.
(2) Location of the story: The film is set in the United States of America.
(3) Similar films in theme or style: "Chinatown" shares thematic elements with "The World Is Not Enough."

6 ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant No.92370204) National Key R&D Program of China (Grant No.2023YFF0725001), Guangdong Basic and Applied Basic Research Foundation (Grant No.2023B1515120057), Guangzhou-HKUST(GZ) Joint Funding Program (Grant No.2023A03J0008, Grant No.2024A03J0630), Education Bureau of Guangzhou Municipality.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale Simple Question Answering with Memory Networks. *arXiv preprint arXiv:1506.02075* (2015).

[2] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NeurIPS*. 2787–2795.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language Models Are Few-shot Learners. *Advances in Neural Information Processing Systems* 33 (2020), 1877–1901.

[4] Sarath Chandar, Sungjin Ahn, Hugo Larochelle, Pascal Vincent, Gerald Tesauro, and Yoshua Bengio. 2016. Hierarchical Memory Networks. *arXiv preprint arXiv:1605.07427* (2016).

[5] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2023. A Survey on Evaluation of Large Language Models. *ACM Transactions on Intelligent Systems and Technology* (2023).

[6] Jiashun Cheng, Man Li, Jia Li, and Fugee Tsung. 2023. Wiener Graph Deconvolutional Network Improves Graph Self-supervised Learning. (2023).

[7] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General Language Model Pretraining with Autoregressive Blank Infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 320–335.

[8] Denis Emelin, Daniele Bonadiman, Sawsan Alqahtani, Yi Zhang, and Saab Mansour. 2022. Injecting domain knowledge in language models for task-oriented dialogue systems. *arXiv preprint arXiv:2212.08120* (2022).

[9] Taoran Fang, Yunchao Zhang, Yang Yang, Chunping Wang, and Lei Chen. 2023. Universal Prompt Tuning for Graph Neural Networks. In *NeurIPS*.

[10] Martin Grohe. 2020. Word2vec, Node2vec, Graph2vec, X2vec: Towards a Theory of Vector Embeddings of Structured Data. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 1–16.

[11] Aditya Grover and Jure Leskovec. 2016. Node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[12] Wenzhong Guo, Jianwen Wang, and Shiping Wang. 2019. Deep Multimodal Representation Learning: A Survey. *Ieee Access* 7 (2019), 63373–63394.

[13] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *Advances in neural information processing systems* 30 (2017).

[14] X. Han, S. Cao, X. Lv, Y. Lin, Z. Liu, M. Sun, and J. Li. 2018. openke: An open toolkit for knowledge embedding. In *EMNLP*.

[15] Xiaotian Han, Zhimeng Jiang, Ninghao Liu, and Xia Hu. 2022. G-mixup: Graph data augmentation for graph classification. In *International Conference on Machine Learning*. PMLR, 8230–8248.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[17] Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. 2022. Graphmae: Self-supervised Masked Graph Autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 594–604.

[18] Linmei Hu, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, and Juanzi Li. 2023. A Survey of Knowledge Enhanced Pre-trained Language Models. *IEEE Transactions on Knowledge and Data Engineering* (2023).

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely Connected Convolutional Networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4700–4708.

[20] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding Via Dynamic Mapping Matrix. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*. 687–696.

[21] M. Kazemi and D. Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *NeurIPS*.

[22] Thomas N Kipf and Max Welling. 2016. Semi-supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

[23] Sotiris B Kotsiantis, Ioannis Zaharakis, P Pintelas, et al. 2007. Supervised Machine Learning: A Review of Classification Techniques. *Emerging Artificial Intelligence Applications in Computer Engineering* 160, 1 (2007), 3–24.

[24] Kundan Krishna, Jeffrey Bigham, and Zachary C Lipton. 2021. Does pretraining for summarization require knowledge transfer? *arXiv preprint arXiv:2109.04953* (2021).

[25] T. Lacroix, N. Usunier, and G. Obozinski. 2018. Canonical Tensor Decomposition for Knowledge Base Completion. In *ICML*.

[26] Hanxiao Liu, Yuexin Wu, and Yiming Yang. 2017. Analogical Inference for Multi-relational Embeddings. In *International conference on machine learning*. PMLR, 2168–2178.

[27] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods In Natural Language Processing. *Comput. Surveys* 55, 9 (2023), 1–35.

[28] Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, et al. 2023. Summary of Chatgpt-related Research and Perspective Towards the Future of Large Language Models. *Meta-Radiology* (2023), 100017.

[29] Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345* (2019).

[30] Zemin Liu, Xingtong Yu, Yuan Fang, and Xinming Zhang. 2023. Graphprompt: Unifying Pre-training and Downstream Tasks for Graph Neural Networks. In *Proceedings of the ACM Web Conference 2023*. 417–428.

[31] Yuanfu Lu, Xunqiang Jiang, Yuan Fang, and Chuan Shi. 2021. Learning to Pre-train Graph Neural Networks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 4276–4284.

[32] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent Advances In Natural Language Processing Via Large Pre-trained Language Models: A Survey. *Comput. Surveys* 56, 2 (2023), 1–40.

[33] OpenAI. 2023. Gpt-4 Technical Report. arXiv:2303.08774 [cs.CL]

[34] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An Imperative Style, High-performance Deep Learning Library. *Advances in Neural Information Processing Systems* 32 (2019).

[36] 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 (2020), 5485–5551.

[37] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. 2019. Dropedge: Towards Deep Graph Convolutional Networks on Node Classification. *arXiv preprint arXiv:1907.10903* (2019).

[38] Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2022. Multitask Prompted Training Enables Zero-shot Task Generalization. In *ICLR*.

[39] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The Graph Neural Network Model. *IEEE transactions on neural networks* 20, 1 (2008), 61–80.

[40] Dhruv Shah, Błażej Osiński, Sergey Levine, et al. 2023. Lm-nav: Robotic Navigation with Large Pre-trained Models of Language, Vision, and Action. In *Conference on Robot Learning*. PMLR, 492–504.

[41] Yi-Fan Song, Zhang Zhang, Caifeng Shan, and Liang Wang. 2020. Stronger, Faster and More Explainable: A Graph Convolutional Baseline for Skeleton-based Action Recognition. In *proceedings of the 28th ACM international conference on multimedia*. 1625–1633.

[42] Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Heung-Yeung Shum, and Jian Guo. 2023. Think-on-graph: Deep and responsible reasoning of large language model with knowledge graph. *arXiv preprint arXiv:2307.07697* (2023).

[43] Mingchen Sun, Kaixiong Zhou, Xin He, Ying Wang, and Xin Wang. 2022. Gppt: Graph Pre-training and Prompt Tuning to Generalize Graph Neural Networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1717–1727.

[44] Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. Colake: Contextualized language and knowledge embedding. *arXiv preprint arXiv:2010.00309* (2020).

[45] Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. 2023. All In One: Multi-task Prompting for Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining (KDD'23)*. 2120–2131.

[46] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2818–2826.

[47] Alon Talmor and Jonathan Berant. 2018. The Web As a Knowledge-base for Answering Complex Questions. arXiv:1803.06643 [cs.CL]

[48] 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, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-tuned Chat Models. *CoRR* abs/2307.09288 (2023).

[49] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *International conference on machine learning*. PMLR, 2071–2080.

[50] Petar Veličković. 2023. Everything Is Connected: Graph Neural Networks. *Current Opinion in Structural Biology* 79 (2023), 102538.

[51] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. 2017. Graph Attention Networks. *stat* 1050, 20 (2017), 10–48550.

[52] Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. 2024. Knowledge Fusion of Large Language Models. In *ICLR*.

[53] Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2023. Can Language Models Solve Graph Problems In Natural Language? *NeurIPS* (2023).

[54] Jianing Wang, Chengyu Wang, Minghui Qiu, Qiuhui Shi, Hongbin Wang, Jun Huang, and Ming Gao. 2022. Kecp: Knowledge enhanced contrastive prompting for few-shot extractive question answering. *arXiv preprint arXiv:2205.03071* (2022).

[55] Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou. 2014. Link prediction in social networks: the state-of-the-art. *arXiv preprint arXiv:1411.5118* (2014).

[56] Q. Wang, Z. Mao, B. Wang, and L. Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *TKDE* 29, 12 (2017), 2724–2743.

[57] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808* (2020).

[58] Zengzhi Wang, Qiming Xie, Zixiang Ding, Yi Feng, and Rui Xia. 2023. Is Chatgpt a Good Sentiment Analyzer? A Preliminary Study. *arXiv preprint arXiv:2304.04339* (2023).

[59] Z. Wang, J. Zhang, J. Feng, and Z. Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *AAAI*, Vol. 14. 1112–1119.

[60] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned Language Models Are Zero-shot Learners. In *International Conference on Learning Representations*.

[61] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought Prompting Elicits Reasoning In Large Language Models. *Advances in Neural Information Processing Systems* 35 (2022), 24824–24837.

[62] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph Neural Networks In Recommender Systems: a Survey. *Comput. Surveys* 55, 5 (2022), 1–37.

[63] 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 (2020), 4–24.

[64] Jun Xia, Lirong Wu, Jintao Chen, Bozhen Hu, and Stan Z Li. 2022. Simgrace: A simple framework for graph contrastive learning without data augmentation. In *Proceedings of the ACM Web Conference 2022*. 1070–1079.

[65] Han Xie, Da Zheng, Jun Ma, Houyu Zhang, Vassilis N. Ioannidis, Xiang Song, Qing Ping, Sheng Wang, Carl Yang, Yi Xu, Belinda Zeng, and Trishul Chilimbi. 2023. Graph-aware Language Model Pre-training on a Large Graph Corpus Can Help Multiple Graph Applications. (2023), 5270–5281.

[66] Peng Xu, Xiatian Zhu, and David A Clifton. 2023. Multimodal Learning with Transformers: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[67] Pinar Yanardag and SVN Vishwanathan. 2015. Deep Graph Kernels. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1365–1374.

[68] Hongbin Ye, Ningyu Zhang, Shumin Deng, Xiang Chen, Hui Chen, Feiyu Xiong, Xi Chen, and Huajun Chen. 2022. Ontology-enhanced Prompt-tuning for Few-shot Learning. In *Proceedings of the ACM Web Conference 2022*. 778–787.

[69] Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. 2023. Natural Language Is All a Graph Needs. *arXiv preprint arXiv:2308.07134* (2023).

[70] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph Contrastive Learning with Augmentations. *Advances in neural information processing systems* 33 (2020), 5812–5823.

[71] Xingtong Yu, Chang Zhou, Yuan Fang, and Xinming Zhang. 2024. Multigprompt for Multi-task Pre-training and Prompting on Graphs. In *WWW*.

[72] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph Transformer Networks. *Advances in neural information processing systems* 32 (2019).

[73] Denghui Zhang, Zixuan Yuan, Hao Liu, Hui Xiong, et al. 2022. Learning to Walk with Dual Agents for Knowledge Graph Reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 5932–5941.

[74] Jiying Zhang, Xi Xiao, Long-Kai Huang, Yu Rong, and Yatao Bian. 2022. Fine-tuning graph neural networks via graph topology induced optimal transport. *arXiv preprint arXiv:2203.10453* (2022).

[75] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A Survey of Large Language Models. *arXiv preprint arXiv:2303.18223* (2023).

[76] 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 (2020), 57–81.

[77] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022. Learning to prompt for vision-language models. *International Journal of Computer Vision* 130, 9 (2022), 2337–2348.

[78] Zhuangdi Zhu, Kaixiang Lin, Anil K Jain, and Jiayu Zhou. 2023. Transfer Learning In Deep Reinforcement Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

A APPENDIX

A.1 Dataset Descriptions

Extensive comparison of our method with other approaches has been conducted on **23** public and synthetic datasets across **6** different tasks of graph and language domains, including:

Table 5: Statistics of datasets for node classification task.

Dataset	#nodes	#edges	#classes	#labels	#training set	#validation set	#testing set
Cora	2,708	5,429	1,433	7	1,896	542	270
CiteSeer	3,327	9,104	3,703	6	2,329	665	333
Pubmed	19,717	88,648	500	3	13,802	3,943	1,972

Table 6: Statistics of datasets for link prediction task.

Dataset	#entities	#relations	#training set	#validation set	#testing set
FB15K	14,951	1,345	483,142	50,000	59,071
WN18	40,943	18	141,442	5,000	5,000
FB15K237	14,541	237	272,115	17,535	20,466
WN18RR	40,943	11	86,835	3,034	3,134
YAGO3-10	123,182	37	1,079,040	5,000	5,000

Table 7: Statistics of datasets for graph classification task.

Dataset	#size	#classes	#avg.nodes	#avg.edges	#training set	#validation set	#testing set
IMDB-BINARY	1,000	2	19.77	96.53	700	200	100
IMDB-MULTI	1,500	3	13.00	65.94	1050	300	150
REDDIT-BINARY	2,000	2	429.63	497.75	1400	400	200
REDDIT-MULTI-5K	4,999	5	508.52	594.87	3499	1000	500
REDDIT-MULTI-12K	11,929	11	391.41	456.89	8350	2386	1193

A.1.1 Graph Domain. **(i) Node classification task**, in which we focus on three popular citation datasets: CORA, PUBMED, CITESEER (with the statistics reported in Table 5). In each citation graph, its nodes denote documents, edges are citations, and each node feature is represented by the bag-of-words representation used to classify the research topic of papers. **(ii) Link prediction task**, in which we consider five benchmark datasets of knowledge graph [21, 25, 26], including WN18, FB15K, WN18RR, FB15K237 and YAGO3-10 (with the statistics shown in Table 6). In each graph, it comprises a set of edges, each consisting of a head entity, a relation edge, and a tail entity. **(iii) Graph classification task**, in which we use two series of IMDB (IMDB-BINARY, IMDB-MULTI) and REDDIT (REDDIT-BINARY, REDDIT-MULTI-5K, REDDIT-MULTI-12K) datasets [67] (with the statistics written in Table 7). IMDB

series represent relational datasets that consist of the ego-networks of 1,000 actors/actresses who played roles in movies. Meanwhile, REDDIT series stands for graphs corresponding to online discussions on Reddit. In each graph, the nodes represent users, and there exists an edge between them if at least one of them responds to the other's comment.

A.1.2 Language Domain. We re-configure the following language tasks into the question-answer (QA) format. (iv) **Paper classification task**, where we create three synthetic datasets (PC-Cora, PC-CS, PC-PM) based on Cora, CiteSeer, and PubMed. Each dataset contains a group of questions that correspond to the node-wise similarity measured from three aspects: semantic similarity, structural similarity, and degree similarity. In particular, we design the following instruction prompts that guide ChatGPT 3.5-turbo to conduct the above similarity measurements:

```
There are two nodes, node {{node_i}} and node {{node_j}}. Their
structural similarity is {{jaccard_similarity}}. Their feature
similarity is {{feature_similarity}}. Their average degree is
{{degree_similarity}}. Question: Is node {{node_i}} similar to node
{{node_j}}? Please Only Answer: 'Yes' or 'No'.
```

In this way, such multi-aspect consideration enables the LLM to perform CoT-based reasoning more effectively. Note that these synthetic datasets share identical statistics as three node classification datasets. (v) **Question answering task**, in which we prepare two open-domain KBQA datasets: Single-Hop KBQA named Simple Questions (SimpleQA) [1] and Multi-Hop KBQA called ComplexWebQuestions(CWQ) [47]. On the one hand, SimpleQA contains 108,442 simple natural language questions, each of which is paired with a corresponding fact from Freebase [4]. On the other hand, CWQ contains a large set of complex questions that require reasoning over multiple web snippets [47]. For the task above, we constructed the following instruction prompts:

```
Please answer to the prompt {{prompt_content}} based on the following
information:
To answer the above question, we develop several intermediate
reasoning questions:{{resoning_questions_list[i]}}, According to
these questions, the discovered graph knowledge
include:{{graph_knowledge[i]}}.
```

(vi) **Movie classification task**, where we create five synthetic datasets (MC-IB, MC-IM, MC-RB, MC-RM5, MC-RM12) based on IMDB and REDD series. Each dataset includes a set of movie prediction questions, each focusing on predicting the type of movie in which a given actor participates. In particular, we first find all 2-hop neighbors of an actor node and count the number of different categories of movies they participated in, respectively. Then a probability vector containing the neighbors' statistics, together with structural information, is provided to facilitate subsequent LLM reasoning and prompt construction. Similarly, the instruction prompts used here are listed as:

```
Actor {{actor_name}} has neighbors that belong to various classes,
the probability vector is {{probabilities}}, is this actor likely
belong to class {{actor_label}}? Only reply yes or no.
```

Note that these synthetic datasets share identical statistics as five graph classification datasets.

A.2 Implementation Details

We conduct our experiments using PyTorch [35] on a Linux Ubuntu Server with 4 RTX 4090 GPUs, 2 NVIDIA A100 40G GPUs, and 8 NVIDIA Tesla V100 GPUs. For each dataset, we report the mean and standard deviation over ten runs. We use two single-layer LSTMs for G-agent and L-agent, respectively. For each pair of graph and language datasets, we first perform CoT prompting on the ChatGPT 3.5-turbo [33] to decompose each of 1000 QA samples into multiple intermediate reasoning questions. For semantic matching, we incorporate the same embedding model T5-Small [36] to vectorize all graph nodes and questions for all datasets. Then, a graph node would be preserved in a question if their cosine similarity is larger than a pre-defined threshold. Other details of model hyper-parameters are reported in the source code.

A.3 Additional experiments

We conducted additional experiments on large and dynamic graphs to further validate our method's scalability.

Firstly, we used a large dataset, ogbl-wikikg2, which includes over 2 million entities and 500 relation types. Here, TransE and ComplEx models were employed for the link prediction task.

Table 8: Experiment Results on OGBL-wikikg2

Model	Metric	MRR	Hits@1	Hits@3	Hits@10
TransE	w/o ours	26.7	21.3	28.9	35.9
	w/ ours	31.9	28.1	33.4	38.0
ComplEx	w/o ours	38.7	33.4	39.7	48.4
	w/ ours	40.9	34.1	42.9	54.7

Secondly, we tested on two datasets, Wikipedia and Reddit, using a Temporal Graph Network (TGN) for dynamic node classification. The results indicate relative improvements of 1.88% and 2.15% in classification accuracy on Wikipedia and Reddit, respectively.

A.4 Discussions and Future work

A.4.1 Computational overhead and Performance gain. The computational burden brought by RL agents to the backbone is negligible, with the computational time increased by less than 3%, and the parameter size increased by less than 5%. The prompt size can also be adjusted by setting various thresholds.

However, we find limited performance gain in some scenarios, which is related to the adapting ability of the specific foundation model to the inserted prompt.

A.4.2 How difficult it would be to use any random graph and LLM pair for integration? Integrating any random graph with an LLM is complex due to the need to bridge modalities, which varies with each graph's features, size, and density. These factors significantly influence how prompts are crafted and applied.

A.4.3 Future Work. Our current approach uses CoT to segment questions for node-wise semantic alignment in graph data, potentially overlooking the benefits of structural graph information that could enhance reasoning in complex QA tasks. Additionally, it does not account for dynamic changes in node relations, such as evolving real-world facts. Moreover, while modifying prompts is crucial for optimizing LLM outputs, developing an active learning algorithm to automatically adjust prompts could significantly improve response quality. These limitations will be addressed in our future research.