GaCLLM: Graph-aware Convolutional Large Language Model for Recommendation

Anonymous ACL submission

Abstract

Leveraging auxiliary textual data can help with user profiling and item characterization in recommender systems (RSs). However, incomplete user and item descriptions limit the potential of textual information in RSs. To this end, we propose a graph-aware convolutional LLM method, eliciting LLMs to summarize from a high-order interaction graph to generate fine-grained descriptions for users and items. We focus on two challenges in this paper: 1) the incompatibility between structural graph and text-aware LLMs; and 2) the limitation of LLMs' capability for long context. To bridge the gap between graph structures and LLMs, we employ the LLM as an aggregator for graph convolution process, eliciting it to infer the graph-based knowledge iteratively. To mitigate the information overload associated with large-scale graphs, we segment the graph processing into manageable steps, progressively incorporating multi-hop information in a least-to-most manner. Experiments on three real-world datasets demonstrate that our method consistently outperforms state-of-the-art approaches.

1 Introduction

Recommender systems (RSs) are pivotal in delivering personalized services to users for their satisfaction and platform profitability. Traditionally, RSs heavily rely on user-item interaction records (Koren et al., 2009) but face challenges with data sparsity (Sun et al., 2019). Recently, there has been a trend towards utilizing auxiliary textual information for recommendation (Torbati et al., 2023). However, texts with users and items often suffer from incompleteness and bias, with users offering vague self-descriptions and providers giving sparse or strategically biased item descriptions. Such texts negatively impact user profiling and item characterization, hindering accurate recommendations.

To enhance the reliability and completeness of textual descriptions, recent approaches have employed large language models (LLMs) to generate LLM-driven descriptions based on raw contents and task-specific prompt instructions (Zheng et al., 2023; Wu et al.; Liu et al., 2023; Wang et al., 2024b), such as incorporating users' behaviors as supplemental knowledge for retrieval-augmented generation (Du et al., 2024; Liu et al., 2024b). Nevertheless, these methods still suffer from unreliable and inaccurate textual generation due to the lack of collaborative user-item insights and the limited scope of information observed by LLMs.

To this end, inspired by the success of graph convolutional networks (GCNs) (Kipf and Welling, 2016), we propose Graph-aware Convolutional LLM (GaCLLM) to integrate high-order collaborative information from user-item graph to provide more evidence for LLM inference, generating fine-grained descriptions of users and items for recommendation. We focus on two main challenges: 1) the incompatibility between structural graph and text-aware LLMs; and 2) the limitation of LLMs' capability for long context. ***Firstly, text-based LLMs are inherently ill-suited for processing structured graph data.*** Existing methods convert graph data into textual form using templates and sampling strategies (Wang et al., 2023; Wu et al., 2024). However, these methods limit the LLMs' ability to maintain a global perspective on graphs, thereby hindering their full potential in utilizing reasoning skills for graph-based knowledge. ***Secondly, large-scale user-item graphs pose context length limitations for LLM inputs by simply describing them in a textual format.*** Specifically, LLMs often struggle to robustly comprehend information from lengthy contextual inputs, particularly when critical information (e.g., key entities in graphs) is located in the middle (Liu et al., 2024a).

To tackle these challenges, we develop a convolutional inference strategy to integrate high-order relations from the user-item interaction graph into LLMs. To align LLMs with graph structures, we employ the LLM as an aggregator function and maintain a global perspective on graphs. Specifically, the LLM assimilates information from neighboring nodes and ensures layer-by-layer propagation throughout the graph. By leveraging high-order relations in the user-item interaction graph, our method enhances reasoning capabilities for better LLM-driven descriptions. To mitigate the information overload associated with large-scale graphs, we segment the graph processing into manageable steps in a least-to-most (Zhou et al., 2022) manner, iteratively incorporating multi-hop neighbor information to refine each node's (i.e., user or item) description. Therefore, the overload of describing the graph can be segmented into several steps with a drastic reduction of context length for LLMs, alleviating the limitations of lengthy inputs to capture critical information for LLM-driven reasoning. Finally, we fuse these LLM-driven descriptions into behavioral graph embeddings to bridge the gap between text information and structural data in the user-item graph for recommendation. We conduct extensive experiments on multiple real-world datasets to show that our method consistently outperforms state-of-the-art approaches and validate the effectiveness of our proposed strategy.

2 RELATED WORK

2.1 Graph-based Recommendation

Graph-based recommender systems (Kipf and Welling, 2016; Huang et al., 2024; Yan et al., 2024) employ deep neural networks to model user-item interactions within graph structures. LightGCN (He et al., 2020) streamlines GCNs for collaborative filtering with simplicity and effectiveness. Then, many studies use contrastive learning (Yu et al., 2022; Chen et al., 2023), transformer (Wei et al., 2023), neighborhood-structure (Lin et al., 2022), and self-supervised learning (Wu et al., 2021) as enhancement. However, they mainly focus on aggregating node embeddings and fail to extract insights from textual descriptions for recommendation.

2.2 LLM for Recommendation

There is increasing interest in leveraging LLMs in recommender systems (Lyu et al., 2024; Bao et al.). Non-tuning methods (Kuo and Chen, 2023; Senel et al., 2024) assume that LLMs possess recommendation capabilities and use them to produce results directly through prompts (Kang et al., 2023; Zhang et al., 2023) and in-context learning (Hou et al., 2024; Wang and Lim, 2024). The tuning paradigm (Lu et al., 2024) employs LLM as feature extractors for downstream tasks, aiming to capture contextual information for a precise understanding of user profiles (Zheng et al., 2023; Du et al., 2024; Ren et al., 2024), user attributes (Wang et al., 2024a), and item descriptions (Liu et al., 2024b). However, relying only on raw text and ignoring graph knowledge leads to hallucinations.

2.3 LLM with Graph Data

Integrating LLMs with graph data (Li et al., 2024; Tang et al., 2024) effectively leverages the rich structure and relationships. Supervised methods use LLMs for graph-aware tasks via encoding text into node embeddings (Chen et al., 2024; Zhang et al., 2021) and incorporating graph elements into training (Sun et al., 2021; Yasunaga et al., 2022; Xie et al., 2023; Zhang et al., 2024b), but they mainly compress graph knowledge into model parameters, overlooking the LLMs' reasoning mechanism. Unsupervised methods (Wang et al., 2023; Andrus et al., 2022; Wu et al., 2024; Zhang et al., 2024a) convert graph information into text via templates or sampling strategies for LLMs to process. However, they lack a global view of the graph and still fail to fully exploit LLMs' reasoning potential.

3 Methodology

3.1 Problem Definition

We denote $\mathcal{U} = \{u_1, \cdots, u_N\}$ and $\mathcal{I} = \{i_1, \cdots, i_M\}$ as the sets of users and items, where N and M are sizes. The interaction records between users and items can be denoted as an interaction matrix $\mathcal{R} \in \mathbb{R}^{N\times M}$ where $\mathcal{R}_{u,i} = 1$ if user u interacted with item i, and 0 otherwise. We also possess the textual information (e.g., user resumes and job descriptions in online recruitment scenarios) of both users, denoted as $\mathcal{T}_u = [w_1, \cdots, w_{l_u}]$ with length l_u for user u, and items, denoted as $\mathcal{T}_i = [w_1, \cdots, w_{l_i}]$ with length l_i for item i, and w_k represents the k-th word. In this paper, our goal is to learn a matching function $g(u, i)$ using the interaction records $\mathcal{R}$ and the textual descriptions. Our task is to recommend K items that a user prefers, as known as top-K recommendation.

3.2 Overview

The overall architecture of GaCLLM is shown in Figure 1. First, we perform supervised fine-tuning (SFT) for LLM to strengthen its effectiveness in

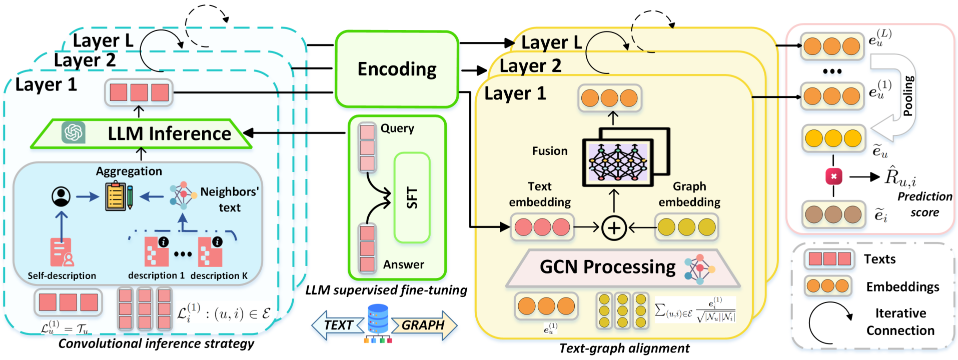


Figure 1: The overall architecture of the proposed method GaCLLM.

the task-related domain. Second, we propose an LLM-based graph-aware convolutional inference strategy to enhance user and item descriptions progressively. Third, we align and integrate the generated text with behavioral information captured through graph-based embeddings. Last, we present the objective function and model learning process.

3.3 Supervised Fine-tuning

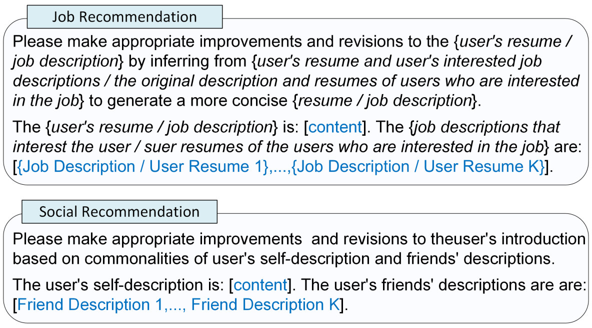
To fully exploit the potential of the LLM in understanding the task-related domain, we begin with fine-tuning it on domain-specific data. This involves the training of the LLM using descriptions from matched user-item pairs, enabling it to learn the alignment between user and item descriptions. Specifically, we employ the prompt template: "**Query**: Given an item's description, generate a user's description that fits it. The item's description is [*Item Desc*]. **Answer**: ", where [*Item Desc*] represents the actual description of the item. The prompt for inferring item descriptions with the provided user description is designed symmetrically. The optimization process involves minimizing the negative log-likelihood loss for these templates, i.e., $\mathcal{L}_{\text{sft}} = -\sum_{k=1}^{|T_{\text{Answer}}|} \log \Pr(w_k \mid w_{<k}, T_{\text{Query}})$, where w_k denotes the k-th word in Answer sentence T_{Answer}, and $\Pr(T_{\text{Answer}}|T_{\text{Query}})$ denotes the generation probability for the produced answer with a given query. This process uses parameter-efficient fine-tuning techniques.

3.4 Convolutional Inference Strategy

Graph Construction. To explore the structured graph with high-order descriptive texts for LLMs, we organize the descriptions of users and items into a unified graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ using the collaborative information among users and items. Specifically, the nodes $\mathcal{V}$ in the graph represent users and items, i.e., $\mathcal{V} = \{u|u \in \mathcal{U}\} \cup \{i|i \in \mathcal{I}\}$. The edges $\mathcal{E}$ are constructed by the interactions between users and items $\mathcal{R} \in \mathbb{R}^{N \times M}$, i.e., $\mathcal{E} = \{(u,i)|\mathcal{R}_{u,i} = 1\}$. Each node in the graph has a textual description, such as a user profile in a social network or a resume of a job seeker.

Least-to-Most Text Enhancement. Recognizing the extensive knowledge, advanced text comprehension, and reasoning capabilities of LLMs, we propose an LLM-based convolutional inference strategy to summarize from a high-order interaction graph to generate fine-grained descriptions for users and items. To make user descriptions more representative, we leverage the LLM to rewrite a user's raw description $\mathcal{T}_u$ by the descriptions of items that the user has interacted with, i.e., $\mathcal{T}'_u = \texttt{LLM}(\texttt{P}_{\texttt{user}}(\mathcal{T}_u, \{\mathcal{T}_i : (u,i) \in \mathcal{E}\}))$, where $\texttt{P}_{\texttt{user}}$ denotes the prompt template for generating user descriptions. Similarly, to enhance item description $\mathcal{T}_i$, we use the LLM to produce the enhanced version considering the descriptions of users by interaction, i.e., $\mathcal{T}'_i = \texttt{LLM}(\texttt{P}_{\texttt{item}}(\mathcal{T}_i, \{\mathcal{T}_u : (u,i) \in \mathcal{E}\}))$, where $\texttt{P}_{\texttt{item}}$ denotes the prompt template for generating item descriptions. The design of a prompt template varies with the tasks. In this paper, we focus on job and social recommendation tasks. The details of the prompts are shown in Figure 2.

To enable LLMs to effectively explore the structured graph, we iteratively use them to refine the descriptions of nodes (users and items) step by step. Specifically, we set the first-layer descriptions of users $\{\mathcal{L}_u^{(1)}|u \in \mathcal{U}\}$ by raw texts provided by users, i.e., $\mathcal{L}_u^{(1)} = \mathcal{T}_u$, and we set the first-layer descriptions of items $\{\mathcal{L}_i^{(1)}|i \in \mathcal{I}\}$ by raw texts given by item providers, i.e., $\mathcal{L}_i^{(1)} = \mathcal{T}_i$. We employ the LLM as an "aggregator" in the graph convolutional process, enhancing its ability to infer graph-based knowledge through iterative steps. The updated



Job Recommendation

Please make appropriate improvements and revisions to the {*user's resume / job description*} by inferring from {*user's resume and user's interested job descriptions / the original description and resumes of users who are interested in the job*} to generate a more concise {*resume / job description*}.

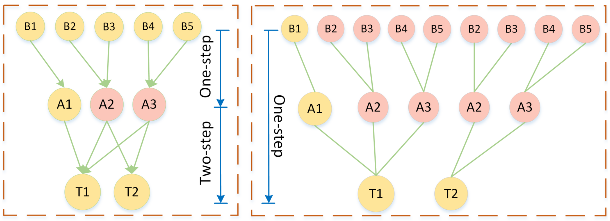
The {*user's resume / job description*} is: [content]. The {*job descriptions that interest the user / suer resumes of the users who are interested in the job*} are: [{Job Description / User Resume 1},...,{Job Description / User Resume K}].

Social Recommendation

Please make appropriate improvements and revisions to theuser's introduction based on commonalities of user's self-description and friends' descriptions.

The user's self-description is: [content]. The user's friends' descriptions are are: [Friend Description 1,..., Friend Description K].

Figure 2: The prompt design for job recommendation (top) and social recommendation (bottom).



Convolutional inference strategy

Plain description strategy

Figure 3: The comparison of token usage of convolutional inference strategy (left) and plain description strategy (right) in text enhancement.

user and item descriptions after each iteration are generated as follows:

$$\mathcal{L}_u^{(l+1)} = \texttt{LLM}(\texttt{P}_{\texttt{user}}(\mathcal{L}_u^{(l)}, \{\mathcal{L}_i^{(l)} : (u,i) \in \mathcal{E}\})), \tag{1}$$

$$\mathcal{L}_i^{(l+1)} = \texttt{LLM}(\texttt{P}_{\texttt{item}}(\mathcal{L}_i^{(l)}, \{\mathcal{L}_u^{(l)} : (u,i) \in \mathcal{E}\})), \tag{2}$$

where $\mathcal{L}_u^{(l+1)}$ and $\mathcal{L}_i^{(l+1)}$ denote the descriptions of users and items at $(l+1)$-th layer after l iterations of generation, capturing l-hop descriptive information within the graph. After L iterations of this LLM-based convolutional inference strategy, we obtain progressively refined descriptions across multiple layers for both users and items.

Token Effectiveness and Efficiency. Compared with organizing all hierarchical node descriptions in the graph structure into a single, *plain* paragraph of prompt (e.g., listing each node and its multi-hop neighbors along with their descriptions), the proposed convolutional inference strategy improves both effectiveness and efficiency in token usage.

First, it optimizes the capture of graph-related information within the limited context length of LLMs. Specifically, the proposed strategy decomposes the ultimate task of description enhancement into multiple steps, where each step (layer) only integrates the descriptions of direct (one-hop) neighbors for the target node. This step-by-step approach effectively alleviates the issues of hallucination and distraction with long inputs, significantly reducing the number of tokens required for each inference.

Second, our convolutional inference strategy efficiently reduces the redundancy in describing the graph for target nodes. Specifically, when comparing the number of nodes required to capture L-hop graph-based information for each node, the proposed method incorporates $O(|\mathcal{G}| \cdot |\mathcal{N}| \cdot L)$ nodes into LLMs, where $|\mathcal{G}|$ denotes the number of nodes in the graph and $|\mathcal{N}|$ denotes the average number of neighbors of each node. In contrast, the *plain* description strategy needs to incorporate $O(|\mathcal{G}| \cdot (1 + \cdots + |\mathcal{N}|^L))$ nodes into LLMs, leading to a significant increase in token usage. Therefore, by minimizing the overlap in node descriptions (the redundant description of common neighbors in Figure 3), our method enhances token efficiency.

3.5 Text-graph Alignment

To bridge the gap between LLM-driven text information and behavioral-based structural data in the user-item graph for recommendation, we propose to align the user and item descriptions with their corresponding graph embeddings in a unified manner. Specifically, the GCN-based embeddings for users and items at the l-th layer, denoted as $\mathbf{e}_u^{(l)}$ and $\mathbf{e}_i^{(l)}$. They can be iteratively updated as follows:

$$\mathbf{e}_u^{(l+1)} = W_l \cdot \left[\sum_{(u,i)\in\mathcal{E}} \frac{\mathbf{e}_i^{(l)}}{\sqrt{|\mathcal{N}_u||\mathcal{N}_i|}} \oplus f(\mathcal{L}_u^{(l)}) \right], \tag{3}$$

$$\mathbf{e}_i^{(l+1)} = W_l \cdot \left[\sum_{(u,i)\in\mathcal{E}} \frac{\mathbf{e}_u^{(l)}}{\sqrt{|\mathcal{N}_u||\mathcal{N}_i|}} \oplus f(\mathcal{L}_i^{(l)}) \right]. \tag{4}$$

where $\mathcal{N}_u$ denotes the set of items that are interacted by user u, and $\mathcal{N}_i$ denotes the set of users that interact with item i. $|\cdot|$ indicates their sizes. We use d to represent the dimension of latent embedding space and $\oplus$ for the fusing function such as concatenation. The matrix $W_l \in \mathbb{R}^{2d\times d}$ denotes the transformation mapping matrix for the l-th layer. In the first layer, each user and item is initialized with a graph embedding based on its ID, represented as $\mathbf{e}_u^{(1)} \in \mathbb{R}^d$ and $\mathbf{e}_i^{(1)} \in \mathbb{R}^d$. To incorporate the textual descriptions associated with users and items, we encode these descriptions into constant text-based embeddings by $f(\cdot)$. In practice, we add a unique token $[CLS]$ before the original text and feed the combined sequence into the *simbert-base-chinese* model. The output of the $[CLS]$ token is used as the semantic embedding for alignment.

To leverage the descriptions of users and items across all layers, we further combine their embeddings from each layer to produce the final embeddings of users and items through mean-pooling:

$$\widetilde{\boldsymbol{e}}_u = \frac{1}{L}\sum\nolimits_{l=1}^{L} \boldsymbol{e}_u^{(l)}; \quad \widetilde{\boldsymbol{e}}_i = \frac{1}{L}\sum\nolimits_{l=1}^{L} \boldsymbol{e}_i^{(l)}. \tag{5}$$

3.6 Objective Function

To measure the matching scores between users and items for final predictions, we propose to compute the inner product of their representations for recommendation prediction scores by $\hat{R}_{u,i} =< \widetilde{\boldsymbol{e}}_u, \widetilde{\boldsymbol{e}}_i >$, where $< \cdot, \cdot >$ denotes the inner product operation for similarity. It produces a score or probability of item i that user u will engage. For the model training process, we use the pairwise loss to define the recommendation objective function as follows:

$$\max_{\Theta} \sum_{(u,i,j)\in\mathcal{D}} \log\sigma(\hat{R}_{u,i} - \hat{R}_{u,j}) - \lambda||\Theta||^2, \tag{6}$$

where the train set $\mathcal{D} = \{(u, i, j)\}$ consists of triplets with a user u, an item i with positive feedback from user u, and an item j with negative feedback from user u. Θ denotes all trainable parameters, and λ is the regularization coefficient of L2 norm $|| \cdot ||^2$.

3.7 Complexity and Applicability

The model parameter of GaCLLM is approximately $\mathcal{O}((M + N) \cdot d + 2 \cdot L \cdot d^2) = \mathcal{O}((M + N) \cdot d)$ as $(M + N) \gg 2 \cdot L \cdot d$. The complexity is similar to the efficient LightGCN (He et al., 2020). As for model training, the time cost is slightly higher than LightGCN due to the additional text embeddings.

For the training phase, LLM-based recommendation methods inevitably require more complexity and training cost than deep learning-based methods. Our method shares similar training and inference costs compared to existing LLM-based methods, e.g., (Zheng et al., 2024; Wu et al., 2024). Notably, the supervised finetuning (Section 3.3) and convolutional inference strategy (Section 3.4) can be done offline and independently with different users, which is not necessary to require more GPU memory for large-scale applications. For text-graph alignment (Section 3.5), the time cost is slightly higher than LightGCN due to the additional text embeddings. The detailed comparisons of complexity and time consumption between GacLLM and LightGCN are summarized in Table 1. For the serving phase, our complexity computation is the same as most of the recommendation methods, e.g., LightGCN. Therefore, the LLM in our method does not change the latency in the serving phase, thus our method is applicable in real-time deployment.

Model	Parameter Number		Time Efficiency	
	Generation	Train	Generation	Train
LightGCN	-	O(M·d)	-	2.27s
Ours	ChatGLM2-6B	O(M·d)	12.28s	3.76s

Table 1: Comparison of Model Performance

Job	# User Resumes	# Job Descriptions	# Interactions
Designs	12,290	9,143	166,270
Sales	15,854	12,772	145,066
Social	# Group A	# Group B	# Connections
Pokec	6,240	6,213	104,152

Table 2: Statistics of datasets.

4 Experiment

4.1 Experimental Setup

Datasets. We investigate two recommendation scenarios. For **job recommendation**, we use two real-world datasets sourced from an online recruiting platform within the ***Design*** and ***Sales*** professions with extensive user-job interactions. The user resumes and job descriptions are available as textual document information. For **social recommendation**, we use a public dataset Pokec Slovakian Social Network (***Pokec***) collected from an online social platform. It contains the friendship relations among users and their self-descriptions. We aim to suggest connections between users based on diverse preferences and attributes. The dataset is divided into subsets ***Pokec-A*** and ***Pokec-B*** by different user groups. The statistics are in Table 2.

Evaluation. We randomly split the dataset equally into training, validation, and test sets. We utilize two well-recognized top-K recommendation metrics, mean average precision ($MAP@K$) and normalized discounted cumulative gain ($NDCG@K$), where K is set to 5 empirically. We run five times and take the average performance as experimental results with different random initializations.

Baselines. We compare our GaCLLM with the following baselines. **Content-based and collaborative filtering RS:** SGPT-BE (Muennighoff, 2022), MF (Koren et al., 2009), and NCF (He et al., 2017). **Graph-based RS:** LightGCN (He et al., 2020), SimGCL (Yu et al., 2022), UltraGCN (Mao et al., 2021), and SGL (Wu et al., 2021). For a fair com-

Models	Job Recommendation				Social Recommendation			
	Design		Sales		Pokec-A		Pokec-B	
	MAP@5	NDCG@5	MAP@5	NDCG@5	MAP@5	NDCG@5	MAP@5	NDCG@5
SGPT-BE	0.0651	0.1042	0.0491	0.0861	0.0724	0.1013	0.0710	0.0980
MF	0.2081	0.3182	0.0957	0.1751	0.2639	0.3838	0.2616	0.3876
NCF	0.2100	0.3258	0.1468	0.2678	0.2969	0.4270	0.2930	0.4273
LightGCN	0.2940	0.4697	0.1658	0.3001	0.3293	0.4664	0.3294	0.4676
SimGCL	0.1471	0.2277	0.0921	0.1658	0.2940	0.4235	0.3093	0.4459
UltraGCN	0.2639	0.4258	0.1469	0.2725	0.3263	0.4691	0.3204	0.4623
SGL	0.2769	0.4418	0.1431	0.2567	0.3047	0.4385	0.3012	0.4394
LLM-CS	0.2669	0.2190	0.1530	0.2803	0.2569	0.3478	0.2527	0.3468
LLM-TES	0.2208	0.3478	0.1520	0.2797	0.2593	0.3517	0.2571	0.3512
LGIR	0.2898	0.4616	0.1694	0.3103	0.3245	0.4390	0.3081	0.4183
RLMRec	0.2816	0.4377	0.1502	0.2641	0.3326	0.4617	0.3238	0.4498
GaCLLM	**0.3060***	**0.4925***	**0.1750***	**0.3234***	**0.3461***	**0.4798***	**0.3446***	**0.4797***
Improvement	4.06%	4.85%	3.32%	4.21%	4.06%	2.28%	4.60%	2.60%

Table 3: Performance of GaCLLM and baseline methods. The best results are in **bold** and the runner-up results are underscored. $*$ indicates significant improvements at the level of 0.05 with a paired t-test.

parison, we enhance graph-based methods with text information to ensure the same utilization of information. **LLM-based RS:** LLM-CS (Chen et al., 2024), LLM-TES (Chen et al., 2024), LGIR (Du et al., 2024), and RLMRec (Ren et al., 2024).

Implementation Details. For the LLM backbone, we use ChatGLM2-6B (Du et al., 2022) for its proficiency in handling multilingual tasks including Chinese, as datasets ***Design*** and ***Sales*** are in Chinese. For the SFT stage, we use LoRA (Hu et al., 2022) with a learning rate of 10^{-5}, LoRA dimension of 128, batch size of 2, 10^4 training steps, and gradient accumulation of 1. To ensure a fair comparison, we fix the embedding size of all methods to 768, batch size to 1024, and regularization coefficient to 10^{-4} with AdamW (Loshchilov and Hutter, 2019) optimizer. Following (Yang et al., 2022; Du et al., 2024), we use 20 negative instances for every target item during evaluation[1].

4.2 Comparison with Baselines

Table 3 shows the overall comparison between GaCLLM and baselines. From the experimental results, we demonstrate that GaCLLM consistently outperforms all baseline methods across all job recommendation and social recommendation scenarios, with average improvements of 4.46%, 3.77%, 3.69%, and 3.60%. Besides, interaction-only (i.e., MF and NCF) and text-only (SGPT-BE) methods show inferior performance compared to the other hybrid approaches, indicating the necessity of utilizing both text and interaction information. In addition, the improvements in GCN-based methods prove the value of extracting both graph and text information for better recommendation outcomes. This supports our motivation to combine LLMs with graph structural information to improve the quality of textual descriptions in recommendation systems. SimGCL shows underwhelming results, likely due to the graphical framework's incompatibility with incorporating text-aware information effectively. Finally, simply adopting the LLM as the encoder (LLM-CS) or zero-shot reasoner (LLM-TES) produces suboptimal performance. LGIR and RLMRec show stronger performance by inferring from direct neighbors but still overlook the more complex, high-order relationships within the graph. As a result, by aligning LLM and high-order graph relations, GaCLLM achieves the best performance, validating its effectiveness.

Models	Design		Sales	
	MAP@5	NDCG@5	MAP@5	NDCG@5
RAW	0.2951	0.4717	0.1692	0.3082
PLAIN	0.2908	0.4655	0.1677	0.3080
w/o-ALIGN	0.2901	0.4654	**0.1753**	0.3212
GaCLLM	**0.3060**	**0.4925**	0.1750	**0.3234**

Models	Pokec-A		Pokec-B	
	MAP@5	NDCG@5	MAP@5	NDCG@5
RAW	0.3402	0.4678	0.3326	0.4655
PLAIN	0.3362	0.4672	0.3287	0.4612
w/o-ALIGN	0.3435	0.4780	0.3331	0.4687
GaCLLM	**0.3461**	**0.4798**	**0.3446**	**0.4797**

Table 4: Performance of ablation variants.

4.3 Ablation Study

To verify the efficacy of the key components of GaCLLM, we test the following variants. **RAW** adopts raw descriptions instead of LLM-driven descriptions by the user-item graph. **PLAIN** removes the convolutional inference strategy, adopting a

[1] Our code is at https://anonymous.4open.science/r/GaCLLM_code-C326.

template to describe all node descriptions related to the target node in a plain way as the inputs of LLMs. **w/o-ALIGN** excludes the alignment with graph embeddings and simply adopts the enhanced descriptions by L-hop neighbors for node embeddings of the L-th layer. Table 4 shows the performance of variants and original GaCLLM. First, the proposed GaCLLM consistently outperforms **RAW** across all scenarios, indicating that utilizing high-order relations in the interaction graph can improve the textual content and thus lead to more accurate recommendation predictions. Second, GaCLLM significantly outperforms **PLAIN**. While **PLAIN** struggles to effectively capture the structured graph by describing high-order relations in a single prompt, GaCLLM elicits the reasoning capacity of LLMs more effectively through a step-by-step, graph-based convolutional inference process. This allows GaCLLM to better utilize the graph structure for improved recommendations and avoids context length limits. Third, GaCLLM outperforms **w/o-ALIGN** as the alignment of textual and graphical representations bridges the gap between LLM-driven information and behavioral patterns. Thus, we can fully leverage the layered descriptions generated by the LLM for recommendation. As such, the ablation study supports the efficacy of GaCLLM and the underlying motivations presented in this paper.

4.4 In-depth Analysis

In this subsection, we further conduct experiments to analyze the impact of hyper-parameters, the supervised fine-tuning step, and the LLM model selection. We also illustrate the effectiveness of our GaCLLM by both quantitative subgroup analysis and qualitative case study.

Number of Layers. In Figure 4, we observe that the best performance is produced by $(4, 3, 2, 2)$ layers for datasets. For real-world applications, we suggest using the grid search on optimal layer numbers for GaCLLM implementation empirically. In addition, Figure 7 shows that deeper layers can capture or generate richer textual information, which is detailed in the Subgroup Analysis.

Supervised Fine-tuning Study. In Figure 5 (left), we evaluate the variant without supervised fine-tuning in Section 3.3. Using *Designs* dataset as an example, we notice a limited improvement, which indicates that the overall performance boost by GaCLLM is **not** obtained directly from the SFT, but from the LLM-based convolutional inference strategy and embedding alignment. Though the impact of SFT is not significant, some recommendation scenarios may contain extra domain-specific information beyond pre-trained knowledge. Thus, SFT step contributes to the adaptability of GaCLLM.

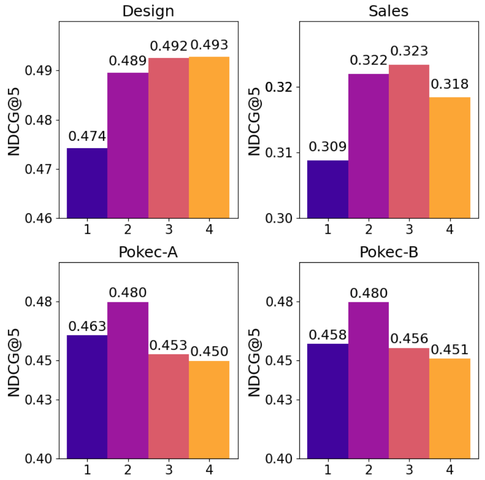


Figure 4: GaCLLM with varying numbers of layers.

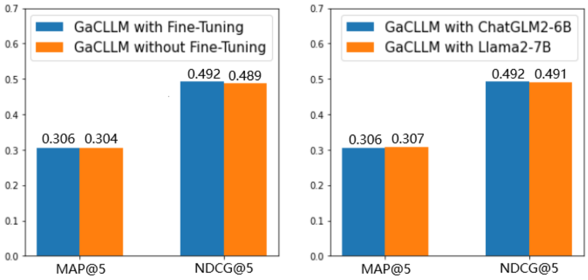


Figure 5: Impact of SFT stage (left) and varying LLM backbones (right) in *Designs* dataset.

LLM Backbone. In Figure 5 (right), we assess GaCLLM using Llama-2-7B as the backbone replacement of the original ChatGLM2-6B with a similar scale. The result shows comparable performance, validating the robustness of our method and the stability of our convolutional inference strategy in description enhancement for recommendation.

Text Encoder. To bridge the gap between LLM-driven text information and behavioral-based graph embeddings, we employ *simbert-base-chinese* to encode user and item text information into latent space. In Table 5, we also explore using other LLM's backbone as text encoder. The results show

Encoder	MAP@5	NDCG@5
simbert-base-chinese	0.3060	0.4925
ChatGLM2-transformer	0.2722	0.4291

Table 5: Performance of the proposed method with varying text encoders in *Designs* dataset.

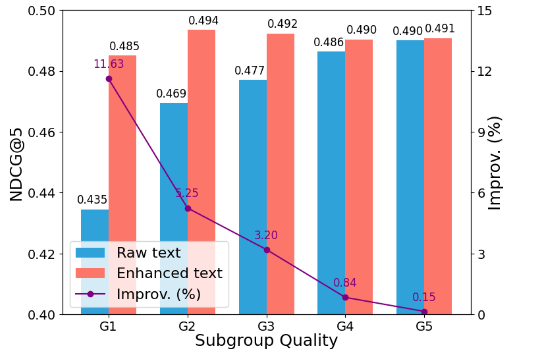


Figure 6: Performance across user subgroups for description improvement analysis.

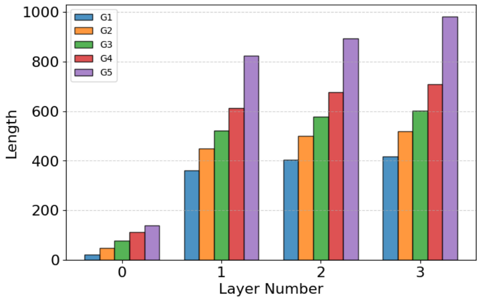
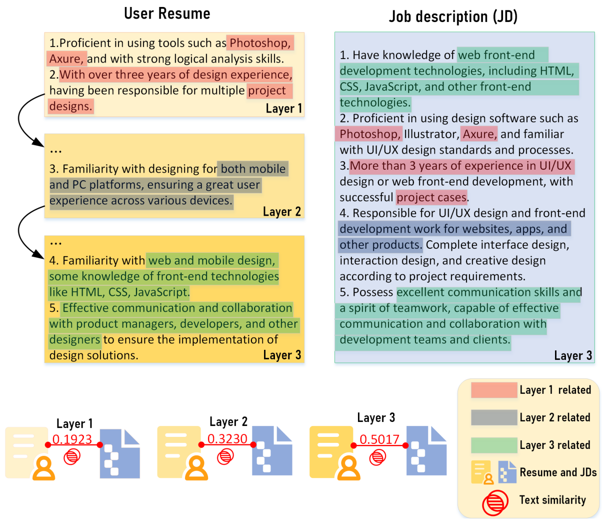


Figure 7: Length of descriptions across layer numbers.

that ChatGLM2 yields suboptimal results as a text encoder, likely due to its decoder-only structure optimized for text generation rather than understanding. For better performance and parameter efficiency, the encoder-only *simbert-base-chinese* is a more suitable choice.

Subgroup Analysis. To investigate how and to what extent our method can enhance the descriptions of users and items, we divided users into five equally sized groups (G1 to G5) based on the length of raw descriptions. The difference between GaCLLM and ***RAW*** in Figure 6 shows the significance of refining descriptions for all raw text. Notably, GaCLLM achieves more substantial improvements in groups with less comprehensive descriptions, highlighting the effectiveness of the LLM-based convolutional inference strategy by leveraging the graph structure. In addition, we also investigate the average lengths of LLM-generated descriptions of each layer for each group as shown in Figure 7. As the layer number increases, the average length rises across all groups, with G1 showing the most significant relative increases. This suggests that deeper layers can capture or generate richer textual information, leveraging whose representations can improve recommendation quality.

Case study. We qualitatively show the efficacy


User Resume

1.Proficient in using tools such as Photoshop, Axure, and with strong logical analysis skills.
2.With over three years of design experience, having been responsible for multiple project designs.
Layer 1

...
3. Familiarity with designing for both mobile and PC platforms, ensuring a great user experience across various devices.
Layer 2

...
4. Familiarity with web and mobile design, some knowledge of front-end technologies like HTML, CSS, JavaScript.
5. Effective communication and collaboration with product managers, developers, and other designers to ensure the implementation of design solutions.
Layer 3

Job description (JD)

1. Have knowledge of web front-end development technologies, including HTML, CSS, JavaScript, and other front-end technologies.
2. Proficient in using design software such as Photoshop, Illustrator, Axure, and familiar with UI/UX design standards and processes.
3.More than 3 years of experience in UI/UX design or web front-end development, with successful project cases.
4. Responsible for UI/UX design and front-end development work for websites, apps, and other products. Complete interface design, interaction design, and creative design according to project requirements.
5. Possess excellent communication skills and a spirit of teamwork, capable of effective communication and collaboration with development teams and clients.
Layer 3

Figure 8: Case study in *Design* dataset.

of the convolutional inference strategy in Figure 8. We highlight contents relevant to the target job from a user's resume across layers. The raw resume contains some relevant information and some irrelevant words. As layers increase, our method progressively refines the resume, removing irrelevant content and focusing on job-specific details. The text similarity between the user's resume and the job description significantly increases by the third layer, showing the LLM's success in reasoning over graph structure. By revising vague information and inferring potential requirements for job matching, we achieve better recommendation outcomes.

5 Conclusion

In this paper, we propose GaCLLM to enhance auxiliary textual information through user-item interactions for recommendation. Our approach bridges the gap between text-based LLMs and graph-based multi-hop relations that contain collaborative information. By employing an iterative convolutional inference strategy, GaCLLM enables efficient propagation of textual information across the graph within constrained token limits to achieve quality improvement. We further align the LLM-driven texts and the behavioral graph embeddings to enhance recommendation performance. Extensive experiments show that GaCLLM consistently outperforms various baseline methods, with ablation studies and in-depth analysis further validating our model design. In future work, we aim to explore using LLMs to handle multi-modality information beyond text for more fine-grained RSs.

Limitation

The primary constraints of this paper are as follows: (1) The training phase requires substantial computational resources for LLM inference. Since some users and items may share similar collaborative information, it may not be necessary to make exact inferences for all nodes in the graph. (2) In real-world scenarios, users often exhibit dynamic preferences for items. However, GaCLLM relies on a static graph, which fails to capture the dynamic preferences underlying users' sequential behaviors. To this end, we leave the exploration of more efficient and dynamic solutions for sequential recommendation as future work.

References

Berkeley R Andrus, Yeganeh Nasiri, Shilong Cui, Benjamin Cullen, and Nancy Fulda. 2022. Enhanced story comprehension for large language models through dynamic document-based knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 10436–10444.

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*, page 1007–1014.

Mengru Chen, Chao Huang, Lianghao Xia, Wei Wei, Yong Xu, and Ronghua Luo. 2023. Heterogeneous graph contrastive learning for recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM)*, page 544–552.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, and Jiliang Tang. 2024. Exploring the potential of large language models in learning on graphs. *SIGKDD Explor. Newsl.*, page 42–61.

Yingpeng Du, Di Luo, Rui Yan, Xiaopei Wang, Hongzhi Liu, Hengshu Zhu, Yang Song, and Jie Zhang. 2024. Enhancing job recommendation through llm-based generative adversarial networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 8363–8371.

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 (ACL)*, pages 320–335.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval (SIGIR)*, pages 639–648.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web (WWW)*, pages 173–182.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *European Conference on Information Retrieval (ECIR)*, pages 364–381.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*.

Zhen Huang, Zhongchuan Sun, Jiaming Liu, and Yangdong Ye. 2024. Group-aware graph neural networks for sequential recommendation. *Information Sciences*, 670:120623.

Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do llms understand user preferences? evaluating llms on user rating prediction. *arXiv preprint arXiv:2305.06474*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Hui-Chi Kuo and Yun-Nung Chen. 2023. Zero-shot prompting for implicit intent prediction and recommendation with commonsense reasoning. In *Findings of the Association for Computational Linguistics (ACL)*, pages 249–258.

Yuhan Li, Zhixun Li, Peisong Wang, Jia Li, Xiangguo Sun, Hong Cheng, and Jeffrey Xu Yu. 2024. A survey of graph meets large language model: Progress and future directions. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI)*, pages 8123–8131.

Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *Proceedings of the ACM Web Conference (TheWebConf)*, pages 2320–2329.

Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024a. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics (TACL)*, 12:157–173.

Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2024b. Once: Boosting content-based recommendation with both open- and closed-source large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM)*, page 452–461.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

Wensheng Lu, Jianxun Lian, Wei Zhang, Guanghua Li, Mingyang Zhou, Hao Liao, and Xing Xie. 2024. Aligning large language models for controllable recommendations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8159–8172.

Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Chris Leung, Jiajie Tang, and Jiebo Luo. 2024. LLM-rec: Personalized recommendation via prompting large language models. In *Findings of the Association for Computational Linguistics: NAACL*, pages 583–612.

Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. Ultragcn: ultra simplification of graph convolutional networks for recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM)*, pages 1253–1262.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Representation learning with large language models for recommendation. In *Proceedings of the ACM on Web Conference 2024*, pages 3464–3475.

Lütfi Kerem Senel, Besnik Fetahu, Davis Yoshida, Zhiyu Chen, Giuseppe Castellucci, Nikhita Vedula, Jason Ingyu Choi, and Shervin Malmasi. 2024. Generative explore-exploit: Training-free optimization of generative recommender systems using LLM optimizers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5396–5420.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.

Zhu Sun, Qing Guo, Jie Yang, Hui Fang, Guibing Guo, Jie Zhang, and Robin Burke. 2019. Research commentary on recommendations with side information: A survey and research directions. *Electronic Commerce Research and Applications (ECRA)*, 37:100879.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. 2024. Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 491–500.

Ghazaleh Haratinezhad Torbati, Anna Tigunova, and Gerhard Weikum. 2023. Unveiling challenging cases in text-based recommender systems. In *Proceedings of the 3rd Workshop Perspectives on the Evaluation of Recommender Systems (Perspectives@RecSys)*.

Lei Wang and Ee-Peng Lim. 2024. The whole is better than the sum: Using aggregated demonstrations in in-context learning for sequential recommendation. In *Findings of the Association for Computational Linguistics (NAACL)*, pages 876–895.

Yan Wang, Zhixuan Chu, Xin Ouyang, Simeng Wang, Hongyan Hao, Yue Shen, Jinjie Gu, Siqiao Xue, James Y Zhang, Qing Cui, et al. 2023. Enhancing recommender systems with large language model reasoning graphs. *arXiv preprint arXiv:2308.10835*.

Yan Wang, Zhixuan Chu, Xin Ouyang, Simeng Wang, Hongyan Hao, Yue Shen, Jinjie Gu, et al. 2024a. Llmrg: Improving recommendations through large language model reasoning graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, pages 19189–19196.

Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Yanbin Lu, Xiaojiang Huang, and Yingzhen Yang. 2024b. RecMind: Large language model powered agent for recommendation. In *Findings of the Association for Computational Linguistics (NAACL)*, pages 4351–4364.

Yinwei Wei, Wenqi Liu, Fan Liu, Xiang Wang, Liqiang Nie, and Tat-Seng Chua. 2023. Lightgt: A light graph transformer for multimedia recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, page 1508–1517.

Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval (SIGIR)*, pages 726–735.

Likang Wu, Zhaopeng Qiu, Zhi Zheng, Hengshu Zhu, and Enhong Chen. 2024. Exploring large language model for graph data understanding in online job recommendations. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAQI)*, pages 9178–9186.

"Xuansheng Wu, Huachi Zhou, Yucheng Shi, Wenlin Yao, Xiao Huang, and year = "2024" Ninghao Liu". "could small language models serve as recommenders? towards data-centric cold-start recommendation". In *"Proceedings of the ACM Web Conference (TheWebConf)*, pages "3566–3575".

Han Xie, Da Zheng, Jun Ma, Houyu Zhang, Vassilis N. Ioannidis, Xiang Song, Qing Ping, et al. 2023. Graph-aware language model pre-training on a large graph corpus can help multiple graph applications. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 5270–5281.

Surong Yan, Chongyang Li, Haosen Wang, Bin Lin, and Yixian Yuan. 2024. Feature interactive graph neural network for kg-based recommendation. *Expert Systems with Applications*, 237:121411.

Chen Yang, Yupeng Hou, Yang Song, Tao Zhang, Ji-Rong Wen, and Wayne Xin Zhao. 2022. Modeling two-way selection preference for person-job fit. In *Proceedings of the 16th ACM Conference on Recommender Systems (RecSys)*, pages 102–112.

Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy S Liang, and Jure Leskovec. 2022. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*, pages 37309–37323.

Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval (SIGIR)*, pages 1294–1303.

Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *arXiv preprint arXiv:2305.07001*.

Mengmei Zhang, Mingwei Sun, Peng Wang, Shen Fan, Yanhu Mo, Xiaoxiao Xu, Hong Liu, Cheng Yang, and Chuan Shi. 2024a. Graphtranslator: Aligning graph model to large language model for open-ended tasks. In *Proceedings of the ACM Web Conference (TheWebConf)*, page 1003–1014.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2021. Greaselm: Graph reasoning enhanced language models. In *International conference on learning representations (ICLR)*.

Yichi Zhang, Zhuo Chen, Lingbing Guo, Yajing Xu, Wen Zhang, and Huajun Chen. 2024b. Making large language models perform better in knowledge graph completion. In *Proceedings of the 32nd ACM International Conference on Multimedia (MM)*, page 233–242.

Zhi Zheng, Wenshuo Chao, Zhaopeng Qiu, Hengshu Zhu, and Hui Xiong. 2024. Harnessing large language models for text-rich sequential recommendation. In *Proceedings of the ACM on Web Conference 2024*, pages 3207–3216.

Zhi Zheng, Zhaopeng Qiu, Xiao Hu, Likang Wu, Hengshu Zhu, and Hui Xiong. 2023. Generative job recommendations with large language model. *arXiv preprint arXiv:2307.02157*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations (ICLR)*.