Exploring Large Language Model for Graph Data Understanding in Online Job Recommendations

Likang Wu[1,2], **Zhaopeng Qiu**[2], **Zhi Zheng**[1,2], **Hengshu Zhu**[2] *, **Enhong Chen**[1] *

[1] University of Science and Technology of China
[2] Career Science Lab, BOSS Zhipin
{wulk,zhengzhi97}@mail.ustc.edu.cn, {zhpengqiu,zhuhengshu}@gmail.com, cheneh@ustc.edu.cn

Abstract

Large Language Models (LLMs) have revolutionized natural language processing tasks, demonstrating their exceptional capabilities in various domains. However, their potential for graph semantic mining in job recommendations remains largely unexplored. This paper focuses on unveiling the capability of large language models in understanding behavior graphs and leveraging this understanding to enhance recommendations in online recruitment, including promoting out-of-distribution (OOD) applications. We present a novel framework that harnesses the rich contextual information and semantic representations provided by large language models to analyze behavior graphs and uncover underlying patterns and relationships. Specifically, we propose a meta-path prompt constructor that aids LLM recommender in grasping the semantics of behavior graphs for the first time and design a corresponding path augmentation module to alleviate the prompt bias introduced by path-based sequence input. By facilitating this capability, our framework enables personalized and accurate job recommendations for individual users. We evaluate the effectiveness of our approach on comprehensive real-world datasets and demonstrate its ability to improve the relevance and quality of recommended results. This research not only sheds light on the untapped potential of large language models but also provides valuable insights for developing advanced recommendation systems in the recruitment market. The findings contribute to the growing field of natural language processing and offer practical implications for enhancing job search experiences.

Introduction

Online recruitment recommendations aim to suggest relevant job opportunities to job seekers based on their preferences and qualifications, improving the chances of matching the right employment. With the exponential growth of online recruitment platforms and the need for efficient and personalized job search experiences, the development of effective job recommendation systems has become crucial.

In online recruitment systems, job postings and resumes are written in natural language. Traditional approaches have treated job-resume matching as a supervised text-matching problem using paired data for training (Qin et al. 2018; Shen et al. 2018). However, online recruitment platforms often suffer from sparse interaction data, with job postings attracting only a few candidates on average (Ramanath et al. 2018). To address this, recent studies (Bian et al. 2020; Yang et al. 2022) have explored the use of behavior graphs to capture high-order interactions and alleviate the sparse interaction issue. These behavior graphs leverage message passing to enhance the understanding of user preferences.

Unlike many general recommendation tasks, it is easy to find that textual understanding forms the backbone of job recommendation, and behavior modeling contributes to the personalized module. In our work, we strive to overcome the accuracy limitations of job recommenders by enhancing the semantic richness of textual representations. Inspired by several recent successful recommendations based on text pre-training (Wu et al. 2023), we introduce a large language model (LLM) as the foundational framework for job recommendation that directly generates targets. Adopting this approach is not only beneficial but also intuitive. For instance, out-of-distribution items usually appear in recruitment markets since new job demands are constantly emerging, such as prompt engineers for generative models. This issue is more complex than traditional cross-domain tasks (Zhao et al. 2023; Jiang et al. 2023; Yu et al. 2023). The powerful semantic mining ability and extensive external knowledge of LLMs augment the generation and associative power of recommenders, which is able to generate reasonable recommendation results for the hard OOD items.

However, the existing learning schema of LLM recommender cannot understand the non-textual behavior graph which weakens the personalized recommendation ability for different job seekers. To tackle this challenge, we propose a meta-path prompt constructor to encode the interaction information of graph into the natural language prompt. Specifically, in such a heterogeneous behavior graph, each meta-path composed of various types of nodes and edges can be transferred into a description naturally since each type indicates a specific and meaningful interaction, e.g., interview, conversation, etc. Along this line, for each job seeker, the LLM captures the high-order interaction feature to augment her personality with the meta-path prompt.

Based on the above analysis, we explore the inclusion of graph data understanding in large language model-based

*Corresponding Author.
†During the BOSS Zhipin internship period.

recommendations for the first time. An efficient large language model named GLRec (Graph-understanding LLM Recommender) is proposed to optimize the recommended quality of job recommendation, which is fine-tuned with LoRa (Hu et al. 2021) on our constructed instruction dataset for aligning the gap between pre-trained knowledge and actual recruitment domain. Especially, our exploration presents two valuable and important findings that largely influence the graph understanding strategy of LLM: (i). Different paths would present different weights for the model decision. (ii). The position bias of the order of path prompts brings unstable answers. For these issues, we carefully design path shuffling, adaptive path selector, and their hybrid path augmentation mechanism to mitigate the adverse effects posed by varying path prompts. The main contributions could be summarized as follows:

- To our best knowledge, we are the first to implement the fine-tuned large language model as job recommender, which promotes matching accuracy via the semantic richness and massive knowledge of LLM.
- We propose the meta-path prompt constructor that leverages LLM recommender to comprehend behavior graphs for the first time and design a corresponding path augmentation module to alleviate the prompt bias.
- We conduct sufficient experiments on real-world recruitment datasets, and the experimental results and visualization cases show the superiority of our model.

Related Work

Job Recommendation

Job Recommendation has been extensively studied in the literature (Kenthapadi, Le, and Venkataraman 2017). Early methods handled this problem (Lu, El Helou, and Gillet 2013) relying on collaborative filtering assumptions. Existing research focused more on text-matching technology. They have been proposed to encode job and resume information. For example, (Shen et al. 2018) utilized CNN for encoding, while (Qin et al. 2018) leveraged RNN and BiLSTM to capture sequential information. (Yan et al. 2019) introduced a profiling memory to learn latent preference representation by interacting with both job and resume. (Luo et al. 2019) explored the effectiveness of adversarial training for job-resume matching. In addition to the aforementioned research, some researchers considered multi-granularity interactions. The ranking-based loss can be used to capture multi-level interactions as supervision signals (Le et al. 2019). (Fu et al. 2021) proposed a bilateral multi-behavior sequence model to describe users' dynamic preferences. These approaches highlighted the importance of considering various interaction patterns and incorporating additional user information to improve the quality of job recommendations. However, online recruitment platforms frequently encounter challenges due to sparse interaction data, resulting in job postings attracting only a limited number of candidates on average (Ramanath et al. 2018). Recent studies (Bian et al. 2020; Yang et al. 2022) have investigated the utilization of behavior graphs to capture high-order interactions and alleviate the problem of sparse interactions.

Large Language Models for Recommendation

LLMs can extract high-quality textual features and use external knowledge to improve recommenders. A review by (Wu et al. 2023) categorized LLM-based recommendation systems into discriminative and generative models. Discriminative models align pre-trained models like BERT with domain-specific data through fine-tuning. (Qiu et al. 2021; Wu et al. 2021a) proposed a pre-training and fine-tuning approach to learn user representations, leveraging content-rich domains to complement users' sparse behavior data. Additionally, some research explored training strategies like prompt tuning. (Penha and Hauff 2020) leveraged BERT's Masked Language Modeling (MLM) head to uncover its understanding of item genres using cloze-style prompts. Prompt4NR (Zhang and Wang 2023) pioneered the application of the prompt learning paradigm for news recommendation. Generative models usually translate recommendation tasks as natural language tasks, and then apply techniques such as in-context learning (Hou et al. 2023; Dai et al. 2022), prompt tuning (Kang et al. 2023; Bao et al. 2023), and instruction tuning (Zhang et al. 2023; Cui et al. 2022) to adapt LLMs to directly generate the recommendation results. Compared to discriminative models, generative models have better natural language generation capabilities. In the recruitment area, there was a generative model which developed LLM with RLHF to generate potential JDs for more explainable recommendations (Zheng et al. 2023). However, despite their successes, LLM recommenders have a glaring limitation: they lack the ability to comprehend graph data, which impedes their potential for personalized adaptation.

Methodology

Preliminary

Problem Formulation Consider a set of candidates $C = \{c_1, c_2, \ldots, c_{n_1}\}$ and a set of jobs $\mathcal{J} = \{j_1, j_2, \ldots, j_{n_2}\}$, where n_1 and n_2 represent the total number of candidates (job seekers) and jobs, respectively. Each candidate and job are associated with textual documents that describe their resumes and job requirements. They are also linked to a collection of directed interaction records (such as interviewing and discussing) within the recruitment platform. These interactions are formally represented as $\mathcal{A}c_i = \{c_i \rightarrow j' | c_i \in C, j' \in \mathcal{J}\}$ and $\mathcal{A}j_k = \{j_k \rightarrow c' | j_k \in \mathcal{J}, c' \in C\}$, indicating the directed interactions initiated by candidate c_i or employer j_k (referred to as a job). Our objective is to predict the compatibility between a job posting and a candidate.

Generative Large Language Models Generative LLMs are powerful language models capable of generating coherent and contextually relevant text. Models like GPT-3 and GPT-4 are trained on vast amounts of text data, enabling them to produce human-like text in response to a given prompt or input. Fine-tuning is a common adaption strategy to align the target of pre-trained model and domain-specific applications, such as two popular paradigms of prompt tuning, and instruction tuning. For all these tuning methods, they have an equal final objective loss of autoregressive

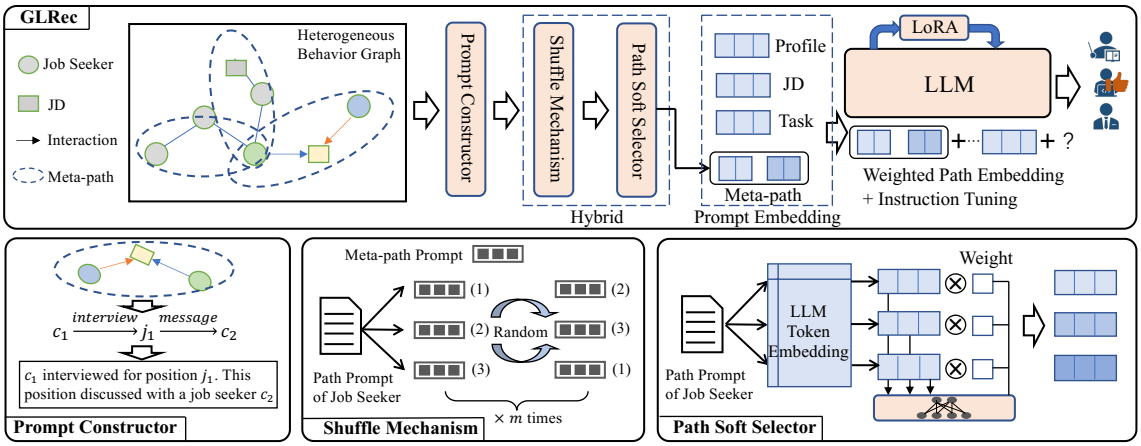


Figure 1: The framework of GLRec for job recommendation.

training as follows:

$$\mathcal{L}_f = \max_{\Theta} \sum_{(x,y)\in\mathcal{T}} \sum_{t=1}^{|y|} \log\left(\mathcal{P}_\Theta\left(y_t \mid x, y_{<t}\right)\right), \quad (1)$$

Taking instruction tuning as an example, which designs and constructs instruction data to restrict the output scope and format. x and y represent the "Instruction Input" and "Instruction Output" in the self-instruct data, respectively, e.g., *Instruction Input: "Do you like this item?", Instruction Output: "Yes."*. And y_t is the t-th token of the y, $y_{<t}$ represents the tokens before y_t, Θ is the original parameters of LLM, and $\mathcal{T}$ is the training set.

Task-specific Instruction In our work, we design two job recommendation tasks to test the LLM recommender following existing related work (Bao et al. 2023), i.e., point-wise and pair-wise job matching. Here we introduce our designed template for the sample in our dataset, where information related to privacy and business has been filtered. Assume there is a job seeker called candidate whose Candidate Profile Prompt and recommended JD Prompt are defined as:
Candidate Profile Prompt: *Age: 25, Education: Bachelor's degree, Graduation School: XXX University, Major: Computer Applied Science, Work Experience: 2 years.*
JD Prompt: *Position: Full Stack Engineer, Educational Requirement: Bachelor's degree, Work Experience: 1-3 years, Skill Requirements: HTML/JAVA/Spring Boot/SQL.*
For the point-wise task, we let the LLM recommender learn to predict the satisfaction of a candidate with a recommended job. The instruction is designed as:
Point-wise Instruction: *You are a recommender, determining whether a candidate would be satisfied with the recommended job. Please answer with "Yes." or "No.".*
For the pair-wise task, we let the LLM recommender learn to justify the preference of a candidate for a recommended job pair. Given two jobs' JD Prompt "A" and "B", the instruction is designed as:
Pair-wise Instruction: *You are a recommender, determining which position will match the candidate. Please answer with "[A]." or "[B].".*

With the above-designed prompts, LLM is able to adapt to a domain recommendation situation. Note that, to ensure the stability of training, we append the JD prompt to the end of the ground truth to increase the predicted length. To further fuse interaction knowledge, as shown in Figure 1, we will illustrate the understanding part of graph data for LLM: behavior meta-path prompt generation.

Behavior Meta-path Prompt Generation

To equip LLM with the ability to comprehend interactive relationships in graph data, we propose a meta-path-based prompt constructor to obtain prompt inputs that represent local subgraphs. Before delving into the details of our approach, it is necessary to provide a formal introduction to heterogeneous graph and meta-path (Wu et al. 2021b).

Definition 1. ***Heterogeneous Graph.*** *$\mathcal{G} = (V, E)$, consists of an object set V and a link set E. $\mathcal{G}$ is also associated with a node type mapping function $\phi : V \to \mathcal{V}$ and a link type mapping function $\psi : E \to \mathcal{E}$. $\mathcal{V}$ and $\mathcal{E}$ denote the sets of predefined object types and link types, where $|\mathcal{V}| + |\mathcal{E}| > 2$.*

Definition 2. ***Meta-path.*** *A meta-path P is defined as a path in the form of $\mathcal{V}_1 \xrightarrow{\mathcal{E}_1} \mathcal{V}_2 \xrightarrow{\mathcal{E}_2} \cdots \xrightarrow{\mathcal{E}_l} \mathcal{V}_{l+1}$ (abbreviated as $\mathcal{V}_1\mathcal{V}_2\cdots\mathcal{V}_{l+1}$), which describes a composite relation $\mathcal{E}_1 \circ \mathcal{E}_2 \circ \cdots \circ \mathcal{E}_l$ between objects $\mathcal{V}_1$ and $\mathcal{V}_{l+1}$, where $\circ$ denotes the composition operator on relations.*

Heterogeneous graphs are more diverse and complex in terms of their semantics compared to homogeneous graphs. Meta-paths are commonly used techniques to mine and represent the interaction semantics within them. In the context of online recruitment, the interactions between job seekers and job positions, which involve different types of behaviors, form a behavior graph. This behavior graph is a typical heterogeneous graph, where different node types include Candidate, JD, and different edge types include messaging, interviewing, matching, and more.

Due to the unique and defined semantics of each type of edge in the behavior graph, it is natural to consider transferring the graph data format meta-path to a natural language description which is acceptable for the large language

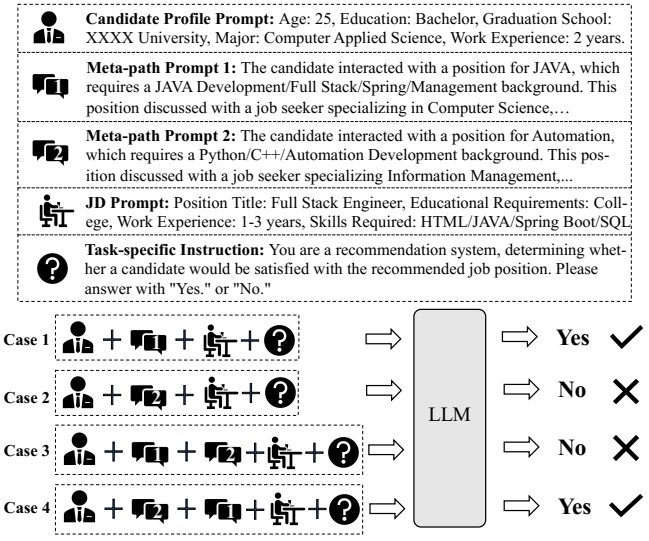


Figure 2: The real cases of path weight and position bias of meta-path prompt input for LLM.

model. We only need to predefine the prompt template according to the appeared edges in a path and then fill in the template with the resume or job description information. For instance, given a typical meta-path $c_1 \xrightarrow{interview} j_1 \xrightarrow{message} c_2$. The prompt template is constructed as:
Meta-path Prompt: c_1 *interviewed for position* j_1. *This position discussed with a job seeker* c_2.

The node information, i.e., the keywords of descriptions of candidates or JD (keywords can be extracted via LLM to compress context length), then will be filled in the meta-path prompt template to generate the final prompt data in our dataset. We add padding to keep the length of each path consistent. The real case can be referred to in Figure 2. In addition, to avoid too similar meta-paths leading to redundancy, we define a simple similarity metric as follows,

$$\mathcal{S}_{i,j} = \frac{|P_i \cap P_j|}{|P_i \cup P_j|}, \quad P_i, P_j \in \Phi_P, \tag{2}$$

where Φ_P denotes the set of sampled paths for a candidate. P_i, P_j indicates two meta-paths in Φ_P. $|P_i \cap P_j|$ is the number of tokens that exist simultaneously in two paths, $P_i \cup P_j$ is the union of them. We ensure that $\mathcal{S}_{i,j} \leq \gamma$ between the final selected M paths and $0 \leq \gamma \leq 1$ is a hyperparameter.

Path Debiasing and Soft Selection Different from the traditional network embedding, sequence-based meta-path prompts would lead to two challenges for LLM to understand the candidates' behavior sub-graph.

Challenge 1. ***Influence of Path Weight****. Different meta-paths would present different weights for the model decision.*

Challenge 2. ***Position Bias of Path Prompt****. The position bias of the order of path prompts brings unstable answers.*

These two challenges appeared when recognizing the pre-trained large language model as a recommender, which hinders the effective modeling of semantic relationships in the graph by LLM recommendation models. To provide a more intuitive explanation, we extracted a real-world case from the log of a popular recruitment platform and visualized them in Figure 2. Specifically, for a job seeker in the IT industry, given his Candidate Profile Prompt, Meta-path Prompt 1, and Meta-path Prompt 2, we further feed the LLM with a Task-specific Instruction belonging to point-wise recommendation. The LLM recommender is expected to output the decision of "Yes" or "No" to present the preference of the candidate. Challenge 1 corresponds to Case 1 and Case 2 in this figure. We can find that the same profile and task description with different behavior meta-paths forces LLM to make different predictions. Obviously, the diversity of technology stacks in Path 1 reveals the candidate's preference for full-stack development, and compared to Path 2, the background of path-related job seeker is more close to our candidate. Therefore, for this candidate, Path 1 is evidently more important for the final decision. For Challenge 2, if we construct the input sequence as Case 3, i.e., the order is meta-path prompt 1 $\rightarrow$ meta-path prompt 2, the LLM outputs the wrong answer "No". But with a reverse path prompt order, the LLM is able to provide an accurate prediction. Similar to the widely known position bias of candidate items (Wu et al. 2023), the position of context prompt clearly misleads the model to generate unstable outputs.

To address the negative impact of these two challenges on the recommendation results, we carefully design an augmentation module specifically for the meta-path prompt, which consists of three concise but effective strategies. The first strategy is **Shuffle Mechanism**. When preparing domain data for the model's supervised fine-tuning (SFT), for each sample that contains multiple paths, we randomly shuffle the meta-path prompts in the sample m times. Here m denotes the conducted times of shuffling. This data augmentation technique allows the model to learn semantic invariance patterns from different combinations of paths, leading to more stable results. It enhances the robustness of the model without introducing redundant information. The second strategy is **Path Soft Selector**. In this work, we regard the path sampling process in Behavior Meta-path Prompt Generation as a hard selection to heuristic selects semantically rich paths. The Path Soft Selector is used to further adaptively assign a learned weight distribution to the constructed meta-path prompts. Firstly, for a given meta-path prompt $\mathcal{M}_i, i \in \{1, 2, ..., M\}$ (M denotes the number of paths), we obtain the LLM word embedding e_t of each token $t \in \mathcal{M}_i$. So, the meta-path embedding H_i of $\mathcal{M}_i$ can be obtained via a mean pooling as follows,

$$H_i = \frac{1}{|\mathcal{M}_i|} \sum_{t \in \mathcal{M}_i} e_t, \quad i \in \{1, 2, ..., M\}. \tag{3}$$

Then we propose a soft selector to calculate the weight for each meta-path embedding as:

$$\alpha_i = \text{softmax}(W_a H_i) = \frac{\exp(W_a H_i)}{\sum_{j=1}^{M} \exp(W_a H_j)}, \tag{4}$$

where $W_a \in \mathcal{R}^{1 \times d_e}$ is a trainable parameter, and d_e denotes the dimension of E_i. To avoid the training collapse

caused by changed value scale, we utilize a controller parameter $\lambda \in (0, 0.5]$ to update word embeddings in Eq. (5).

$$\hat{e}_t = e_t + \lambda \cdot \alpha_i e_t, \quad t \in \mathcal{M}_i, \tag{5}$$

Compared with most existing tuned or non-tuned LLM models, our prompt augmentation mechanism considers phrase-based attention to distinguish different paths. Actually, this simple solution can be transferred to other similar situations, such as weighed sentence embeddings.

What's more, the third strategy is the **Hybrid Mechanism** which implements Shuffle Mechanism and Path Soft Selector simultaneously. This hybrid module is expected to address the both two challenges. We will evaluate these three strategies in the experiment section.

LLM Instruction Tuning and Recommendation

In this subsection, we will introduce the instruction tuning and recommendation process, which aims to align the used LLM with the recommendation task effectively and efficiently. For instruction tuning, we follow the general supervised fine-tuning method to minimize the autoregressive loss calculated by ground truth and corresponding LLM output. In our work, we mask the loss position of the prompt part. Specific prompt format, task-specific instruction, and ground truth have been introduced in the Preliminary section. However, direct fine-tuning of the entire model can be computationally intensive and time-consuming. To address this, we propose a lightweight fine-tuning strategy using LoRA, which involves freezing the pre-trained model parameters and introducing trainable rank decomposition matrices into each layer of the Transformer architecture. This approach facilitates lightweight fine-tuning while reducing GPU memory consumption. And the final learning objective can be computed as follows:

$$\mathcal{L}_f = \max_{\Theta_L} \sum_{(x,y)\in\mathcal{T}} \sum_{t=1}^{|y|} \log\left(P_{\Theta+\Theta_L}\left(y_t \mid e_x, y_{<t}\right)\right), \tag{6}$$

where Θ_L is the LoRA parameters and we only update LoRA parameters during the training process. Note that, different from existing fine-tuning frameworks for recommendation systems, we replace their token input x by the embedding e_x in Eq. (6), since we update the prompt token embedding in the soft selector.

As for the recommendation process, since the trained model has learned the output format of our defined ground truth after several SFT alignment steps. So our designed answer parsing is a simple way. We catch the softmax probability of label generation (the token used to denote label, such as "Yes./No." or "[A]/[B]" in our work) in the position of model's output corresponding to that in the ground truth. Along this line, the final prediction probability is calculated.

Experiments

Experimental Settings

Datasets. We conduct experiments on two datasets RecrX and RecrY with different scales which are collected from a real-world and large online recruitment platform in China to assess recommendation methods. The datasets were constructed from the online logs and contained two kinds of behavior: Match and Interaction, corresponding to the matching set and interaction set mentioned in Problem Formulation. Besides, each candidate (and job) is associated with a descriptive text (i.e., resume or job description). The overall statistics are shown in Table 1. From the statistical data, it can be seen that job recommendation is a sparsely interactive scenario. The segmentation ratio of the training set and testing set is 5:1. Note that all sensitive or private information has been filtered out from the data.

Dataset	# Candidates	# Jobs	# Match	# Interaction
RecrX	12,440	19,318	23,879	54,147
RecrY	18,260	26,576	47,725	119,529

Table 1: The statistics of datasets.

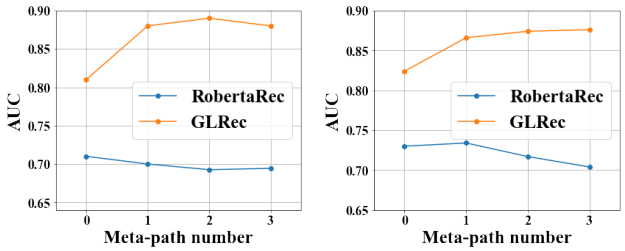
Baseline. To provide a comprehensive evaluation of our GLRec model, we compare it against both LLM-based and related representative job recommendation methods. **RobertaRec** (Liu et al. 2019): Candidate resume and JD text are encoded into fixed-length vectors using RoBERTa and then used to calculate similarity scores, enabling personalized recommendations. **HGT** (Hu et al. 2020): Heterogeneous Graph Transformer is a powerful graph learning model which propagates the embeddings (initialized by RoBERTa) of candidates and jobs on graph to capture high-order interactions. **DPGNN** (Yang et al. 2022): The advanced job recommender Dual-Perspective GNN incorporates two different nodes for each candidate (or job) to model the two-way selection preference. **TALLrec** (Bao et al. 2023): An advanced fine-tuned LLM recommender that uses instruction tuning on self-instruct data with users' historical interactions. We change its backbone to BELLE the same as ours for the Chinese corpus.

Evaluation Metric. We evaluate the two tasks using the conventional metric: Area Under the Receiver Operating Characteristic (AUC), as our two tasks can be transferred to binary classification problems. We do not employ ranking-based metrics because, during the fine-tuning process, the text sequence output of LLM requires ground truth for item order sequences, which, in reality, doesn't exist.

Implementation Details. In this paper, we utilize BELLE-LLaMA-7B (Ji et al. 2023) as the pre-trained LLM backbone due to its expanded Chinese vocabulary. The instruction-tuning and model inference, using LoRa, are conducted on Tesla A100 80G GPUs. To ensure consistent sequence inputs within each batch (batch size is 32), we apply padding to sequences with a maximum length of 512. Our approach incorporates the meta-path prompt and user-specific task instructions as model inputs for personalized recommendations. In our experiments, we investigate the impact of different numbers of paths, specifically $M \in [0, 1, 2, 3]$, for GLRec, and the shuffled times $m = 2$ for $M \geq 2$. In our work, we select paths with 3 nodes because they offer a balance between meaningful

Task	Point-wise						Pair-wise	
Split	Random		OOD_position		OOD_JD		Random	
Dataset	RX	RY	RX	RY	RX	RY	RX	RY
RobertaRec	0.710	0.734	0.503	0.528	0.506	0.536	0.727	0.740
HGT	0.744	0.756	0.572	0.595	0.576	0.593	0.747	0.751
DPGNN	0.727	0.743	0.596	0.603	0.588	0.617	0.744	0.756
TALLrec	0.842*	0.829*	0.770*	0.788*	0.766*	0.798*	0.849*	0.825*
GLRec	**0.891**	**0.876**	**0.810**	**0.843**	**0.814**	**0.852**	**0.905**	**0.883**
Improve ↑	18.4%	14.1%	25.2%	28.3%	26.4%	29.8%	15.5%	13.2%

Table 2: Job recommendation performance of AUC on test set, where * indicates the best result among baselines. Improve ↑ refers to the average enhancement achieved by GLRec in comparison to the baseline models. RX (RY) indicates RecrX (RecrY).



(a) RecrX (b) RecrY

Figure 3: The impact of meta-path number on performance.

semantics and minimal redundancy with the experimental feedback. Further details regarding the path prompt and instructions can be found in the Methodology section. Additionally, both RobertaRec and HGT have a token embedding dimension of 768, and HGT utilizes mean pooling to obtain the initial node embedding. For all methods, we optimize model parameters using the Adam (Kingma and Ba 2014) optimizer with a default learning rate of 1e-4, minimizing the MSE loss as the optimization objective. For the hyperparameters of update controller λ and similarity threshold γ, we set $\lambda = 0.1$ and $\gamma = 0.3$ according to the experimental feedback. We release the code of model[‡].

Performance Comparison

Quantitative Comparison. We conduct quantitative performance experiments on two datasets. As mentioned in the task definition in Section Methodology, the point-wise and pair-wise settings are implemented for evaluation. We also explore the influence of the OOD situation on different models. The experimental split settings of Random, OOD_position, and OOD_JD are introduced below:

- **Random**: We randomly split the training and testing dataset based on the interaction records of each user.
- **OOD_position**: The intersection on JD's "job position" feature between training set and testing set is empty.
- **OOD_JD**: The intersection on JD items between the training set and the testing set is empty.

[‡]https://github.com/WLiK/GLRec

Our experimental results are reported in Table 2. Overall, our proposed GLRec model achieves the best performance among all baselines. There are distinctive score gaps between GLRec and all baselines according to the improvement in Table 2. It demonstrates the superiority and adaptability of the large-scale model framework that incorporates relationship understanding and extensive semantic knowledge in the job recommendation scenario. What's even more exciting is that GLRec demonstrates impressive performance on OOD tasks. While its performance may decline slightly compared to the random setting, our model achieves a significant breakthrough compared to other models, which essentially results in near-random guessing. This phenomenon illustrates the necessity of utilizing knowledge association for model generalization. Going deeper into the part of baselines, the graph-based HGT and DPGNN outperform the conventional dual-tower matching model (RobertaRec) in the context of job recommendation, which further proves the significance of learning relationships. What's more, we find that most models perform better on the pair-wise task than that of point-wise task. That is to say, directly determining whether an item is suitable is more challenging than comparing its priority with another item.

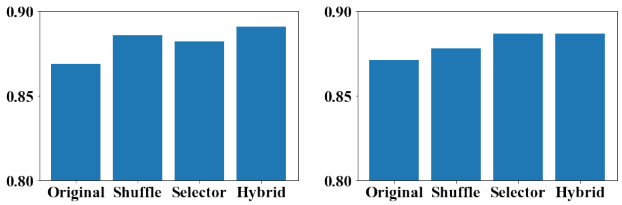
Qualitative Comparison. To give a more intuitive visualization, some qualitative comparison results produced by models are shown in Table 3. Specifically, the first two rows are straightforward, allowing multiple models to predict accurately. In the third row, solely using the user's profile isn't sufficient for prediction. It's crucial to note that the JAVA position (Node 1) the user interacted with aligns well with the target job in skill requirements. Consequently, only TALLrec and GLRec produced correct predictions. The final row emphasizes the significance of higher-order interactions, i.e., path, in LLM recommendations. Although there's a perceived mismatch between the candidate's finance major and the target job, interactions within the testing engineer and fintech sectors provide nuanced hints. For such complex cases, while the TALLrec model, relying on past behaviors, errs, only the GLRec model predicts correctly.

The Impact of Meta-path Number

We investigate the impact of meta-path number on the effectiveness of GLRec. Here we evaluate the point-wise performance on Random setting using AUC for different num-

Candidate	Node 1 (Job)	Node 2 (Job Seeker)	Target Job	GT	Rob	TALL	GLRec
Bachelor's degree, Computer Science, 3 years of work experience, skills...	Front-end Developer, Skill requirements: JavaScript / HTML5 / Vue	Bachelor's degree, Computer Applications Tech, Work experience: 2 years, skills...	Java, Qualification: Bachelor's degree, 5-10 years experience, Skill requirements: Java/System Architecture/Database	**No**	No	No	No
Bachelor's degree, Business Administration, 9 years of work experience, skills...	Project Assistant, Skill requirements: Project Engineering Management	Bachelor's degree, International Economics, 3 years of work experience, skills...	Project Assistant, Qualification: Bachelor's degree, 3 years or more, Skill requirements: Project Engineering Management	**Yes**	Yes	Yes	Yes
Bachelor's degree, Computer Applications Tech, 2 years of experience, skills...	JAVA, Skill requirements: JAVA / Spring / Team Management Experience	Associate's degree, Internet of Things Tech, 4 years of work experience, skills...	Full Stack, Qualification: Associate's degree, 1-3 years of work experience, Skill requirements: JAVA / Spring / HTML	**Yes**	No	Yes	Yes
Bachelor's degree, Finance, 10 years of work experience, skills...	Function Testing, Skill requirements: Software Testing / Requirement Alignment	Bachelor's degree, Financial Engineering, 2 years of work experience, skills...	Test Engineer, Qualification: Bachelor's degree, 3 years of work experience, Skill requirements: Functional/Unit Testing	**Yes**	No	No	Yes

Table 3: Some representative cases of our implemented models in the performance comparison experiment. Node 1 and Node 2 denote the nodes in a sampled meta-path of Candidate. RobRec denotes RobertaRec, and GT denotes Ground Truth.



(a) 2 paths (b) 3 paths

Figure 4: The Impact of Bias of Meta-path Prompt.

bers of meta-paths, ranging from 0 to 3. We also input the meta-path prompt (removing extra instruction text for feature conciseness) into RobertaRec for comparison. From the line graph of Figure 3, we can observe the following trends:

- For GLRec, the results consistently increase as the number of meta-paths increases.
- One notable observation is the significant improvement in GLRec's performance when transitioning from 0 meta-paths to 1 meta-path, and achieving the peak with only 2 or 3 meta-paths. The core increases from 0.71 to 0.88, indicating a substantial boost in recommendation effectiveness. This improvement suggests that the chain-of-thought ability of the LLM, inspired by in-context learning, plays a crucial role in GLRec's performance.
- For RobertaRec, which does not incorporate behavior graph understanding, the values remain relatively stable across different meta-path numbers. The reason is that discriminative BERT-based model lacks the ability to effectively understand prompts like generative LLMs.

The results indicate that the inclusion of behavior graph understanding through meta-path prompt has a significant positive impact on the effectiveness of GLRec.

The Impact of Bias of Meta-path Prompt

Due to the sequential nature of language model input, the construction of multi-path prompt sequences results in a human-induced position bias, or order bias, which disrupts the final decision-making of LLM model. Additionally, this input pattern does not allow the model to learn the importance of semantic information in different paths. Therefore, we design a path shuffle mechanism, a path soft selector, and a hybrid mechanism combining both to enhance the model's understanding of path information and mitigate bias. The experimental results on RecrX are reported in Figure 4. Here the metric is AUC and the task is point-wise setting.

According to Figure 4, our three strategies can all surpass the original input without path prompt augmentation in both two sub-experiments, which proves the necessity of path debiasing. Although the shuffle mechanism and soft selector have their own advantages and disadvantages in two different path scale experiments, both can relatively improve the quality of the results. And the hybrid module of both can bring more stable results, indicating that it is indeed necessary for the model to consider the position factors of input meta-paths and the influencing factors of different path prompts on decision-making in experiments, in order to cope with actual recommendation scenarios. Actually, in other similar scenarios, such as the input for LLM consists of multiple sentence prompts without prior order, our proposed shuffle mechanism and the soft selector can both play a certain role in enhancing the robustness of model training.

Conclusion

In conclusion, we introduced GLRec, a pioneering job recommendation model that seamlessly integrated large language models (LLMs) with behavior graph comprehension. The innovative meta-path prompt constructor effectively translated the intricate interaction details into natural language prompts, thereby refining personalized recommendation strategies. In the testing stage, rigorous evaluations affirmed GLRec's efficacy, highlighting its dominant performance across real-world datasets. This investigation not only propelled the evolution of LLM-centric recommendations but also charted fresh avenues for harnessing graph data in enhancing the personalized capabilities of LLMs.

Acknowledgments

This research was partially supported by grants from National Key Research and Development Program of China (Grant No. 2021YFF0901003).

References

Bao, K.; Zhang, J.; Zhang, Y.; Wang, W.; Feng, F.; and He, X. 2023. TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation. *CoRR*, abs/2305.00447.

Bian, S.; Chen, X.; Zhao, W. X.; Zhou, K.; Hou, Y.; Song, Y.; Zhang, T.; and Wen, J.-R. 2020. Learning to match jobs with resumes from sparse interaction data using multi-view co-teaching network. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 65–74.

Cui, Z.; Ma, J.; Zhou, C.; Zhou, J.; and Yang, H. 2022. M6-Rec: Generative Pretrained Language Models are Open-Ended Recommender Systems. *CoRR*, abs/2205.08084.

Dai, D.; Sun, Y.; Dong, L.; Hao, Y.; Sui, Z.; and Wei, F. 2022. Why Can GPT Learn In-Context? Language Models Secretly Perform Gradient Descent as Meta-Optimizers. *CoRR*, abs/2212.10559.

Fu, B.; Liu, H.; Zhu, Y.; Song, Y.; Zhang, T.; and Wu, Z. 2021. Beyond matching: Modeling two-sided multi-behavioral sequences for dynamic person-job fit. In *Database Systems for Advanced Applications: 26th International Conference, DASFAA 2021, Taipei, Taiwan, April 11–14, 2021, Proceedings, Part II 26*, 359–375. Springer.

Hou, Y.; Zhang, J.; Lin, Z.; Lu, H.; Xie, R.; McAuley, J. J.; and Zhao, W. X. 2023. Large Language Models are Zero-Shot Rankers for Recommender Systems. *CoRR*, abs/2305.08845.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Hu, Z.; Dong, Y.; Wang, K.; and Sun, Y. 2020. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, 2704–2710.

Ji, Y.; Deng, Y.; Gong, Y.; Peng, Y.; Niu, Q.; Ma, B.; and Li, X. 2023. BELLE: Be Everyone's Large Language model Engine. https://github.com/LianjiaTech/BELLE.

Jiang, J.; Zhao, H.; He, M.; Wu, L.; Zhang, K.; and Fan, J. 2023. Knowledge-Aware Cross-Semantic Alignment for Domain-Level Zero-Shot Recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 965–975.

Kang, W.; Ni, J.; Mehta, N.; Sathiamoorthy, M.; Hong, L.; Chi, E. H.; and Cheng, D. Z. 2023. Do LLMs Understand User Preferences? Evaluating LLMs On User Rating Prediction. *CoRR*, abs/2305.06474.

Kenthapadi, K.; Le, B.; and Venkataraman, G. 2017. Personalized Job Recommendation System at LinkedIn: Practical Challenges and Lessons Learned. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Le, R.; Hu, W.; Song, Y.; Zhang, T.; Zhao, D.; and Yan, R. 2019. Towards effective and interpretable person-job fitting. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1883–1892.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lu, Y.; El Helou, S.; and Gillet, D. 2013. A recommender system for job seeking and recruiting website. In *Proceedings of the 22nd International Conference on World Wide Web*.

Luo, Y.; Zhang, H.; Wen, Y.; and Zhang, X. 2019. Resumegan: An optimized deep representation learning framework for talent-job fit via adversarial learning. In *Proceedings of the 28th ACM international conference on information and knowledge management*, 1101–1110.

Penha, G.; and Hauff, C. 2020. What does BERT know about books, movies and music? Probing BERT for Conversational Recommendation. In *RecSys*, 388–397. ACM.

Qin, C.; Zhu, H.; Xu, T.; Zhu, C.; Jiang, L.; Chen, E.; and Xiong, H. 2018. Enhancing person-job fit for talent recruitment: An ability-aware neural network approach. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, 25–34.

Qiu, Z.; Wu, X.; Gao, J.; and Fan, W. 2021. U-BERT: Pre-training User Representations for Improved Recommendation. In *AAAI*, 4320–4327. AAAI Press.

Ramanath, R.; Inan, H.; Polatkan, G.; Hu, B.; Guo, Q.; Ozcaglar, C.; Wu, X.; Kenthapadi, K.; and Geyik, S. C. 2018. Towards deep and representation learning for talent search at linkedin. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2253–2261.

Shen, D.; Zhu, H.; Zhu, C.; Xu, T.; Ma, C.; and Xiong, H. 2018. A joint learning approach to intelligent job interview assessment. In *IJCAI*, volume 18, 3542–3548.

Wu, C.; Wu, F.; Yu, Y.; Qi, T.; Huang, Y.; and Xie, X. 2021a. Userbert: Contrastive user model pre-training. *arXiv preprint arXiv:2109.01274*.

Wu, L.; Li, Z.; Zhao, H.; Liu, Q.; Wang, J.; Zhang, M.; and Chen, E. 2021b. Learning the implicit semantic representation on graph-structured data. In *Database Systems for Advanced Applications: 26th International Conference, DASFAA 2021, Taipei, Taiwan, April 11–14, 2021, Proceedings, Part I 26*, 3–19. Springer.

Wu, L.; Zheng, Z.; Qiu, Z.; Wang, H.; Gu, H.; Shen, T.; Qin, C.; Zhu, C.; Zhu, H.; Liu, Q.; et al. 2023. A Survey on Large Language Models for Recommendation. *arXiv preprint arXiv:2305.19860*.

Yan, R.; Le, R.; Song, Y.; Zhang, T.; Zhang, X.; and Zhao, D. 2019. Interview Choice Reveals Your Preference on the Market: To Improve Job-Resume Matching through Profiling Memories. In *Proceedings of the 25th ACM SIGKDD*

International Conference on Knowledge Discovery & Data Mining.

Yang, C.; Hou, Y.; Song, Y.; Zhang, T.; Wen, J.-R.; and Zhao, W. X. 2022. Modeling Two-Way Selection Preference for Person-Job Fit. In *Sixteenth ACM Conference on Recommender Systems*.

Yu, Y.; Liu, Q.; Wu, L.; Yu, R.; Yu, S. L.; and Zhang, Z. 2023. Untargeted attack against federated recommendation systems via poisonous item embeddings and the defense. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 4854–4863.

Zhang, J.; Xie, R.; Hou, Y.; Zhao, W. X.; Lin, L.; and Wen, J. 2023. Recommendation as Instruction Following: A Large Language Model Empowered Recommendation Approach. *CoRR*, abs/2305.07001.

Zhang, Z.; and Wang, B. 2023. Prompt Learning for News Recommendation. *arXiv preprint arXiv:2304.05263*.

Zhao, C.; Zhao, H.; Li, X.; He, M.; Wang, J.; and Fan, J. 2023. Cross-Domain Recommendation via Progressive Structural Alignment. *IEEE Transactions on Knowledge and Data Engineering*.

Zheng, Z.; Qiu, Z.; Hu, X.; Wu, L.; Zhu, H.; and Xiong, H. 2023. Generative Job Recommendations with Large Language Model. arXiv:2307.02157.