Knowledge Graph Retrieval-Augmented Generation for LLM-based Recommendation

Shijie Wang[1], Wenqi Fan[1]*, Yue Feng[2]*, Shanru Lin[3], Xinyu Ma[4], Shuaiqiang Wang[4], Dawei Yin[4],
[1]The Hong Kong Polytechnic University, [2]University of Birmingham,
[3]City University of Hong Kong, [4]Baidu Inc,
shijie.wang@connect.polyu.hk; wenqifan03@gmail.com; y.feng.6@bham.ac.uk; lllam32316@gmail.com;
xinyuma2016@gmail.com; shqiang.wang@gmail.com; yindawei@acm.org

Abstract

Recommender systems have become increasingly vital in our daily lives, helping to alleviate the problem of information overload across various user-oriented online services. The emergence of Large Language Models (LLMs) has yielded remarkable achievements, demonstrating their potential for the development of next-generation recommender systems. Despite these advancements, LLM-based recommender systems face inherent limitations stemming from their LLM backbones, particularly issues of hallucinations and the lack of up-to-date and domain-specific knowledge. Recently, Retrieval-Augmented Generation (RAG) has garnered significant attention for addressing these limitations by leveraging external knowledge sources to enhance the understanding and generation of LLMs. However, vanilla RAG methods often introduce noise and neglect structural relationships in knowledge, limiting their effectiveness in LLM-based recommendations. To address these limitations, we propose to retrieve high-quality and up-to-date structure information from the knowledge graph (KG) to augment recommendations. Specifically, our approach develops a retrieval-augmented framework, termed **K-RagRec**, that facilitates the recommendation generation process by incorporating structure information from the external KG. Extensive experiments have been conducted to demonstrate the effectiveness of our proposed method.

1 Introduction

Recommender systems, as techniques designed to assist people in making decisions in their daily lives, are increasingly gaining impact in various fields (Kenthapadi et al., 2017; He et al., 2020; Fan et al., 2019), such as online shopping, job matching, and social media. Recently, Large Language Models (LLMs) have achieved significant breakthroughs, which further drive developments in various domains (Fan et al., 2024b; Zhao et al., 2024; Wu et al., 2023a). Especially with the success of LLMs, recommender systems have seen rapid growth (Geng et al., 2022; Bao et al., 2023; Qu et al., 2024). By training on a wide range of data, LLMs (e.g., GPT-4 (Achiam et al., 2023) and LLaMA (Touvron et al., 2023)) are able to acquire extensive knowledge and demonstrate exceptional language understanding capability. This capability enables LLM-based recommender systems to capture user preferences through a nuanced understanding of relevant attributes (e.g., user profiles, item descriptions, historical interactions) for more accurate recommendations. As a result, LLM-based recommender systems have emerged as a new paradigm in recommendation technology (Zhao et al., 2024).

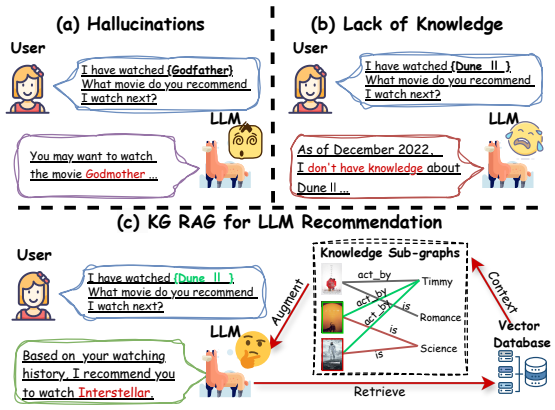


Figure 1: Illustration of the issues of hallucinations and lack of domain-specific knowledge in LLM-based recommender systems and how they can be addressed by knowledge graph retrieval-augmented generation (KG RAG).

However, despite their powerful language understanding and generalization capability, LLM-based recommender systems face significant challenges, including hallucinations and lack of up-to-date and domain-specific knowledge (Luo et al., 2023). Specifically, one key issue is that LLM-

*Corresponding authors

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 27152–27168
July 27 - August 1, 2025

based recommender systems may generate recommendations that are entirely fictional due to the inherent limitations of LLMs. For example, an LLM-based recommender system may recommend *"Godmother"*, a non-existent film, to the user who has watched *"The Godfather"*, as illustrated in Figure 1 (a). Additionally, LLMs usually lack up-to-date knowledge, which can prevent them from recommending the latest films or products in a timely manner. As illustrated in Figure 1 (b), the LLM-based recommender system is unable to recommend the latest films due to the training data only containing up to December 2022. Furthermore, LLMs often lack domain-specific knowledge, as recommendation-oriented corpora are very limited during the training phase of LLMs (Geng et al., 2022). Consequently, LLMs may struggle to meet the nuanced needs of recommendation tasks. To alleviate these issues, one potential solution is to frequently fine-tune the LLMs with up-to-date and domain-specific knowledge. However, the massive parameters of LLMs make this process computationally expensive and time-consuming, which severely hinders the practical application in the real world.

More recently, Retrieval-Augmented Generation (**RAG**) leveraging external databases to provide specific knowledge shows promise to solve these problems (Fan et al., 2024a; Gao et al., 2023). By incorporating an external knowledge base, RAG can retrieve relevant and up-to-date information to complement the LLM's inherent knowledge, thereby mitigating the issues of hallucinations and knowledge gaps (Khandelwal et al., 2019; Min et al., 2020; Li et al., 2024). This makes RAG a promising technique for enabling LLMs to provide effective recommendations without the need for costly fine-tuning (Ram et al., 2023).

Despite this potential, vanilla RAG methods that rely on documents and paragraphs often introduce unnecessary noise and even harmful disturbance, which can negatively impact the accuracy and reliability of recommendations (He et al., 2024). In addition, the structural relationships between entities are overlooked in typical RAG, resulting in the sub-optimal reasoning capability of LLM-based recommender systems (Luo et al., 2023). To address the limitations, a prospective solution is to incorporate structured knowledge such as *items' knowledge graph (KG)* to help improve recommendation performance. Specifically, KGs offer structured, factual, and editable representations of knowledge, which can provide a faithful knowledge source for recommendations. As shown in Figure 1 (c), retrieving structured knowledge from the KG can significantly enhance the recommendation capabilities of LLM-based recommender systems.

However, it is challenging to effectively retrieve KGs to enhance the recommendation capabilities of LLMs. First, KGs store rich factual knowledge in a structured format. Simply retrieving the triplets or first-order neighbors of an entity (i.e., item) with semantic information neglects the importance of these higher-order neighborhood effects among entities/items, resulting in sub-optimal recommendation performance. Second, indiscriminate retrieving for each item, regardless of whether the retrieval is necessary or the content is relevant, can degrade the performance of the recommendation while severely reducing the model's efficiency (Asai et al., 2023; Labruna et al., 2024). Furthermore, structured data in KGs is typically encoded for LLM in the form of serialized text (Wu et al., 2023b; Sun et al., 2023), which is insufficient for fully exploiting the structural information inherent in the data (Perozzi et al., 2024; Fatemi et al., 2023). Therefore, it is crucial to explore more effective and expressive ways of representing structured data, allowing LLMs to effectively leverage the structure information of retrieved knowledge sub-graphs for recommendations.

To address the aforementioned challenges, in this paper, we propose a knowledge retrieval-augmented recommendation framework, namely **K-RagRec**, to provide up-to-date and reliable knowledge by retrieving relevant knowledge from item's KGs for recommendation generation in the era of LLMs. Specifically, our proposed framework first performs knowledge sub-graph indexing on the items' KG at a coarse and fine granularity to construct the knowledge vector database. Next, a popularity selective retrieval policy is designed to determine which items should be retrieved, followed by the retrieval of specific sub-graphs from the knowledge vector database. To refine the quality of retrieval and ensure the most relevant results are prioritized at the top of the input, we subsequently re-rank the retrieved knowledge sub-graphs. Finally, we introduce a GNN and projector to align the retrieved knowledge sub-graphs into the semantic space of the LLM for knowledge-augmented recommendation. The main contributions of this paper can be summarized as follows:

• We propose a novel framework that retrieves

faithful knowledge from KGs to augment the recommendation capability of LLM. Note that we introduce a flexible indexing method for KGs, which can provide a comprehensive view of a node's neighborhood within KG.
- We design a popularity selective retrieval strategy to determine whether an item needs to be retrieved based on its popularity, significantly improving efficiency.
- We introduce a more expressive graph encoder for structured data inclusion in LLMs, that can facilitate the LLM to effectively leverage the structure information and avoid long context input.
- We conduct comprehensive experiments on various real-world datasets to evaluate the effectiveness of the proposed K-RagRec framework.

2 Related Work

Recently, RAG has emerged as one of the most representative technologies in the field of generative AI, combining the strengths of retrieval systems and language models (LM) to produce coherent and informative text. Early methods, such as REALM (Guu et al., 2020), RETRO (Borgeaud et al., 2022), and DPR (Karpukhin et al., 2020), typically involve retrieving relevant fragments from a large corpus to guide the LM generation. However, standard RAG methods often struggle to accurately retrieve all relevant textual chunks, due to unnecessary noise and even harmful disturbance in the documents. To address these limitations, recent studies (Baek et al., 2023; Wu et al., 2023b; He et al., 2024; Luo et al., 2023; Wang et al., 2023; Sen et al., 2023; Sun et al., 2023) focus on retrieving structured and faithful knowledge from graphs for enhancing generations. For example, Retrieve-Rewrite-Answer (Wu et al., 2023b) retrieves relevant sub-graphs from KG and converts retrieved sub-graphs into text for the generation. G-Retriever (He et al., 2024) explores retrieving sub-graphs from various types of graphs to alleviate the hallucinations of LLM.

With the explosion of LLMs in recommendations, a few works (Di Palma, 2023; Wu et al., 2024; Ang et al., 2024) make initial explorations in RAG for recommendations. For instance, the work (Di Palma, 2023) proposes leveraging knowledge from movie or book datasets to enhance recommendations. Nevertheless, retrieving faithful structured knowledge from the KGs for recommendations is under-explored and shows great potential. To fill this gap, we propose to retrieve knowledge sub-graphs from KGs to enhance the recommendation performance of LLM. We provide more comprehensive related work on Appendix A.3.

3 Methodology

In this section, we first introduce some key notations and concepts in this work. Then, we provide the details for each component of our proposed framework K-RagRec.

3.1 Preliminary

Knowledge Graph: In this work, we propose to leverage external knowledge databases (i.e., KGs) to augment LLMs for recommendations. KGs contain abundant factual knowledge in the form of a set of triples: $\mathcal{G} = \{(n, e, n') \mid n, n' \in N\ ,\ e \in E\}$, where N and E denote the set of entities and relations, respectively. For example, a triple $Interstellar \xrightarrow{directed\_by} Nolan$ indicates that the movie *"Interstellar"* was directed by the director *"Nolan"*.

LLM-based Recommendations: Let $U = \{u_1, u_2, \ldots, u_n\}$ and $V = \{v_1, v_2, \ldots, v_m\}$ represent the sets of users and items, where n and m are the sizes of users and items, respectively. The goal of an LLM-based recommender system is to understand users' preferences by modeling users' historical items interactions $V^{u_i} = [v_1^{u_i}, v_2^{u_i}, \cdots, v_{|L_{u_i}|}^{u_i}]$ (*e.g.*, clicks, bought, etc.), where $|L_{u_i}|$ is interaction sequence length for user u_i. Notably, item v_j' side information, such as title and description, is publicly available to enhance LLM for modeling user preferences. In our setting, we consider asking the LLM to select user u's preferred item v from the candidate set $C = \{v_j\}_{j=1}^{M}$, where M is the number of candidate items. The candidate set C typically consists of one positive sample as well as $M-1$ negative samples. Specifically, for a frozen LLM f_δ with parameters δ, we denote an input-output sequence pair as (Q, A), where Q is the recommendation query/prompt, which consists of task descriptions and users' historical items. The output A is the LLM's prediction. Furthermore, we introduce the concept of GNNs in appendix A.2.

3.2 The Overview of Proposed Method

As shown in Figure 2, our proposed K-RagRec consists of five crucial components: *Hop-Field Knowledge Sub-graphs for Semantic Indexing, Popularity Selective Retrieval Policy, Knowledge Sub-graphs*

Retrieval, Knowledge Sub-graphs Re-Ranking, and Knowledge-augmented Recommendation. The model first performs indexing of hop-field knowledge sub-graphs within the KG. Following this, a popularity selective retrieval policy is implemented to determine which items should be retrieved or augmented. The model then retrieves specific sub-graphs from the knowledge vector database. Subsequently, the retrieved knowledge sub-graphs are re-ranked to refine the retrieval quality. Finally, the retrieved knowledge sub-graphs are utilized with the original prompt to generate recommendations.

3.3 Hop-Field Knowledge Sub-graphs for Semantic Indexing

Typically, retrieving knowledge from KG chunks the KG into nodes (He et al., 2024) or triplets (Luo et al., 2023) and only retrieves content locally around the target entity in the KG. However, these methods just naively retrieve the first-order neighbors of an entity (i.e., item), which makes it difficult to capture these higher-order neighborhood effects among entities/items in the recommendation process. Therefore, to effectively retrieve knowledge from the KG, we propose performing semantic indexing on hop-field knowledge sub-graphs, which can flexibly chunk KGs and provide a comprehensive view of a node's neighborhood in KG. As illustrated in Figure 2 component 1, we first introduce a pre-trained language model (PLM), such as SentenseBert (Reimers, 2019), to capture the semantic information for node n_o as follows:

$$\mathbf{z}_{n_o} = \text{PLM}(x_{n_o}) \in R^d, \tag{1}$$

where d is the dimension of the output representation. Similarly, we also capture the semantic information for edge/relation e_o in KG:

$$\mathbf{r}_{e_o} = \text{PLM}(x_{e_o}) \in R^d, \tag{2}$$

where x_{n_o} and x_{e_o} are the text attributes (e.g., item's title and descriptions) of node n_o and edge/relation e_o, respectively.

To retrieve nuanced knowledge of both coarse and fine graph structures from KG, we introduce a GNN (i.e., $\text{GNN}_{\phi_1}^{\text{Indexing}}(\cdot)$ with parameters ϕ_1) to aggregate information from neighbors for entities, where the l_{th}-hop embedding $\mathbf{z}_{n_o}^{(l)}$ of a central entity n_o can be defined by:

$$\mathbf{z}_{n_o}^{l} = \text{GNN}_{\phi_1}^{\text{Indexing}}(\{\mathbf{z}_{n_m}^{(l-1)}, \mathbf{r}_{e_{<o,m>}}^{(l-1)} : n_m \in \mathcal{N}(n_o)\}), \tag{3}$$

where $\mathcal{N}(n_o)$ is the set of neighbours of node n_o, and $e_{<o,m>}$ is the edge between node n_o and n_m. For each entity, its l-hop representation can be seen as a knowledge sub-graph representation containing the l-hop neighbors of itself. Therefore, we can express the knowledge sub-graph representation of $g_o \in \mathcal{G}$ as $\mathbf{z}_{g_o}$, where $\mathcal{G}$ is the set containing all the knowledge sub-graphs. For each sub-graph, we store its representation in a *knowledge vector database*.

3.4 Popularity Selective Retrieval Policy

Although RAG can augment the LLM for modeling user preferences with retrieved knowledge, retrieving each item can cost a significant amount of retrieval time, which can severely degrade the user experience and cause user churn. Meanwhile, most users' online behaviours in recommender systems are following the power law distribution (Abdollahpouri et al., 2017; Celma and Herrera, 2008) in which a small proportion (e.g., less than 20%) of items (i.e., popular items) often account for a large proportion of users' online behaviours (e.g., more than 80%), while cold-start items have a few interactions from users. Therefore, most models tend to keep rich knowledge of popular items, resulting in an inferior performance for cold-start items (Zhao et al., 2023). To this end, we design a popularity selective retrieval policy to determine whether an item needs to be augmented from KG based on its popularity (e.g., sales volume and page view). Particularly, the item is retrieved if its popularity is less than the pre-defined threshold p, otherwise not. By incorporating this strategy, the retrieval time in K-RagRec can be significantly reduced to achieve more efficient retrieval.

3.5 Knowledge Sub-graphs Retrieval

Given the query for knowledge sub-graphs retrieval, we adopt the same PLM as the first indexing step to ensure that the query is in the consistent embedding space as the knowledge sub-graph representation. We define the text attribute (e.g., item's title and descriptions) x_{q_j} of the item v_j that needs to be retrieved as the query and obtain its semantic information $\mathbf{q}_j$ as:

$$\mathbf{q}_j = \text{PLM}(x_{q_j}) \in R^d. \tag{4}$$

Next, we retrieve the top-K most similar sub-graphs $G_j = \{g'_1, ..., g'_K\}$ from $\mathcal{G}$ for item v_j:

$$G_j = \text{argtopK}_{g_* \in \mathcal{G}} \, \text{sim}\left(\mathbf{q}_j, \mathbf{z}_{g_*}\right), \tag{5}$$

where $\text{sim}(\cdot, \cdot)$ is a similarity metric for measuring the similarity between the query representation $\mathbf{q}_j$ and knowledge sub-graph g_*'s representation $\mathbf{z}_{g_*}$ in knowledge vector database. Finally, the retrieved

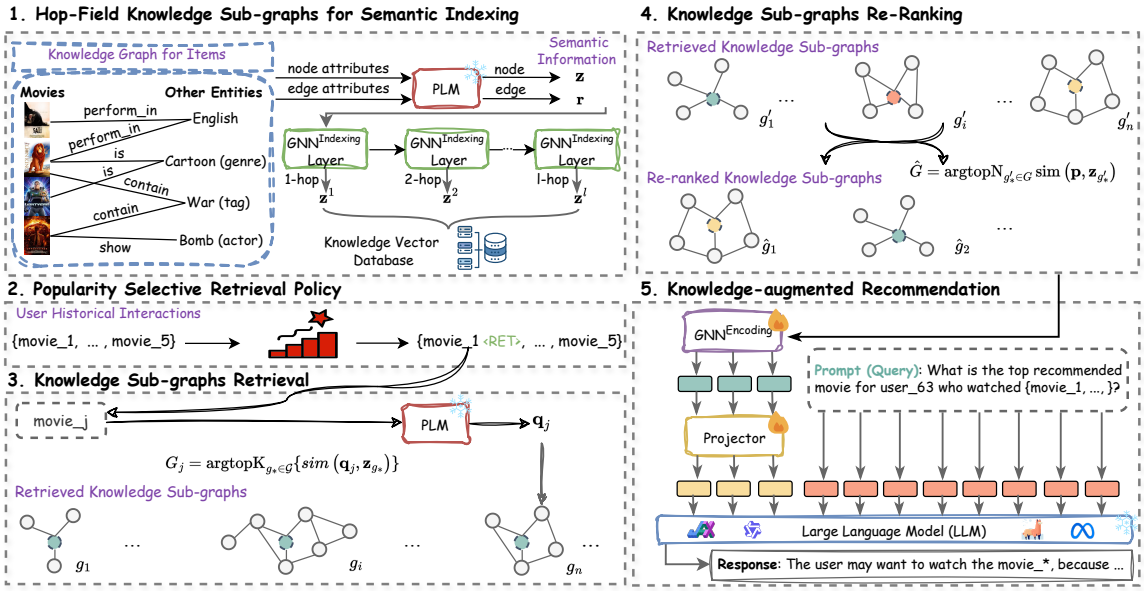


Figure 2: The overview of the K-RagRec. It contains five key components: *Hop-Field Knowledge Sub-graphs for Semantic Indexing, Popularity Selective Retrieval Policy, Knowledge Sub-graphs Retrieval, Knowledge Sub-graphs Re-Ranking, and Knowledge-augmented Recommendation.*

knowledge sub-graphs for items required to be retrieved from user u_i's historical interactions V^{u_i} will be used to form a knowledge sub-graph set G.

3.6 Knowledge Sub-graphs Re-Ranking

To refine the quality of retrieval for enhancing recommendation performance, the next crucial step is knowledge sub-graphs re-ranking. Feeding all retrieved knowledge sub-graphs $g' \in G$ directly into LLM can lead to information overload if the user u_i has a long historical interaction list towards items. Therefore, we execute re-ranking to shorten the retrieved knowledge sub-graphs and ensure the most relevant knowledge sub-graphs are prioritized at the top of the prompt. Specifically, we adopt the recommendation prompt as a query for re-ranking, which consists of **task descriptions** and **users' historical items**. For example, this recommendation prompt can be *"What is the top recommended movie for the user who watched {Matrix, ..., Iron Man}?"*. For consistency, we adopt the same PLM to capture the semantic information of the above prompt as $\mathbf{p}$, and re-rank the knowledge sub-graphs in G to obtain a Top-N knowledge sub-graphs set $\hat{G}$:

$$\hat{G} = \text{argtopN}_{g'_* \in G} \, \text{sim}\left(\mathbf{p}, \mathbf{z}_{g'_*}\right). \tag{6}$$

3.7 Knowledge-augmented Recommendation

To facilitate the LLM's better understanding of the structure of retrieved knowledge sub-graphs and to avoid long contexts, we further integrate another GNN encoder $\text{GNN}^{\text{Encoding}}_{\phi_2}$ with parameter ϕ_2 to enhance the representation learning of structural information:

$$\mathbf{h}_{\hat{g}_*} = \text{GNN}^{\text{Encoding}}_{\phi_2}(\{\hat{g}_* : \hat{g}_* \in \hat{G}\}). \tag{7}$$

An MLP projector MLP_θ with parameter θ is further introduced to shift to mapping all sub-graphs embedding in $\hat{G}$ into the LLM embedding space:

$$\hat{\mathbf{h}}_{\hat{G}} = \text{MLP}_\theta([\mathbf{h}_{\hat{g}_1}; ...; \mathbf{h}_{\hat{g}_N}]), \tag{8}$$

where $[.;.]$ represents the concatenation operation. The extracted knowledge sub-graphs embedding $\hat{\mathbf{h}}_{\hat{G}}$ as the soft prompt is then appended before the input token embedding in LLM.

3.8 Optimization for K-RagRec

The training process can be broadly considered as soft prompt tuning, where the retrieved knowledge sub-graphs are a series of soft graph prompts. Formally, the generation process can be expressed as follows:

$$p_{\delta,\theta,\phi_1,\phi_2}(Y \mid \hat{G}, x_q) = \prod_{k=1}^{r} p_{\delta,\theta,\phi_1,\phi_2}(y_k \mid y_{<k}, \hat{\mathbf{h}}_{\hat{G}}, x_q). \tag{9}$$

Therefore, instead of fine-tuning the LLM model extensively, we only learn parameters of two GNNs (i.e., $\text{GNN}^{\text{Indexing}}_{\phi_1}$, $\text{GNN}^{\text{Encoding}}_{\phi_2}$) and projector MLP_θ, while parameters δ of LLM backbone are frozen. We update the parameters ϕ_1, ϕ_2 and θ through the Cross-Entropy Loss $\mathcal{L}(Y, A)$, where Y is the ground-truth and A is LLM's prediction.

4 Experiment

In this section, we evaluate the effectiveness of our proposed framework through comprehensive experiments. First, we present the experimental settings,

including details about the datasets, compared baselines, evaluation metrics, and the parameter configurations. Then, we report the main experimental results, highlighting the performance of the proposed framework compared with various baseline methods. Finally, we analyze the contributions of individual model components and the impact of parameters used in our framework. We also assess the generalizability of K-RagRec in the zero-shot setting. We present the generalizability study in Appendix A.7.

4.1 Experimental Settings

4.1.1 Datasets

To evaluate the performance of our K-RagRec framework, we adopt three real-world datasets. **MovieLens-1M**[1] is a dataset containing approximately one million movie ratings and textual descriptions of movies (i.e., "title"). **MovieLens-20M**[2] is a large-scale movie ratings dataset encompassing over 20 million ratings from more than 138,000 users on 27,000 movies. **Amazon Book**[3] is a book recommendation dataset that records more than 10 million user ratings of books and the titles of the books. In addition, we adopt the popular knowledge graph *Freebase*[4] and filter out the triples related to the three datasets to reconstruct the KG. The statistics of these three datasets and KG are presented in Table 3.

4.1.2 Baselines

In the realm of LLM-based recommendation research, our work pioneers the investigation of retrieving knowledge from KGs to enhance the recommendation capabilities of LLMs. Therefore, to evaluate the effectiveness, we compare our proposed framework with a series of meticulously crafted KG RAG-enhanced LLM recommendation baselines. We first include two typical inference-only methods Retrieve-Rewrite-Answer (Wu et al., 2023b) (KG-Text) and KAPING (Baek et al., 2023), where the former retrieves sub-graphs and textualizes them, and the latter retrieves triples. We exclude some knowledge reasoning path-based approaches (Luo et al., 2023), as it is difficult to retrieve faithful knowledge reasoning paths solely from the user's interaction items. Next, we compare K-RagRec with various prompt-tuning approaches augmented by retrieval, including Prompt Tuning with KG-Text (PT w/ KG-Text), GraphToken (Perozzi et al., 2024) with retrieval (GraphToken w/ RAG) as well as G-retriever (He et al., 2024). Additionally, we evaluate our method against Lora Fine-tuning with retrieval (Lora w/ KG-Text) (Hu et al., 2021).

4.1.3 Evaluation Metrics

To evaluate the effectiveness of our K-RagRec framework, we employ two widely used evaluation metrics: Accuracy (ACC), and Recall@k (He et al., 2020). We present results for k equal to 3, and 5. Inspired by recent studies (Zhang et al., 2024; Hou et al., 2022), we adopt the *leave-one-out strategy* for evaluation. Specifically, for each user, we select the last item that the user interacted with as the target item and the 10 interaction items prior to the target item as the historical interactions. Then, we leverage LLM to predict the user's preferred item from a pool of 20 candidate items ($M = 20$), which contains one target item with nineteen randomly sampled items. For trained models (including prompt tuning and fine-tuning), we compute Recall@k by extracting the probability assigned to each item and evaluating the model's ability to rank the target item within the top-k predictions. In addition, we also conduct comparison experiments with 10 candidate items ($M = 10$) as shown in Appendix A.5.

4.1.4 Parameter Settings

We implement the framework on the basis of PyTorch and conduct the experiments on 2 NVIDIA A6000-48GB GPUs. We adopt the SentenceBert to encode entities, relations, and query attributes. We use the 3 layers Graph Transformer as the $\text{GNN}^{\text{Indexing}}$ and $\text{GNN}^{\text{Encoding}}$ for MovieLens-1M and 4 layers for MovieLens-20M and Amazon Book. The layer dimension is set to 1024, and the head number is set to 4. The popularity selective retrieval policy threshold p is set to 50%. For each item that needs to be retrieved, we retrieve the top-3 most similar sub-graphs. The re-ranking knowledge sub-graph number N is set to 5. More experiment details are shown in Appendix A.1. We also present several prompt examples in Appendix A.12.

4.2 Overall Performance Comparison

We compare the recommendation performance of K-RagRec with various baselines on three open-

[1]https://grouplens.org/datasets/movielens/1m/
[2]https://grouplens.org/datasets/movielens/20m/
[3]https://jmcauley.ucsd.edu/data/amazon/
[4]https://developers.google.com/freebase

Table 1: Performance comparison of different KG RAG-enhanced LLM recommendations. The best performance and the second-best performance are marked in red and blue, respectively. ACC and R@k denote Accuracy and Recall@k, respectively.

Models	Methods		MovieLens-1M			MovieLens-20M			Amazon Book		
			ACC	R@3	R@5	ACC	R@3	R@5	ACC	R@3	R@5
LLama-2	**Inference-only**	KG-Text (Wu et al., 2023b)	0.076	-	-	0.052	-	-	0.058	-	-
		KAPING (Baek et al., 2023)	0.079	-	-	0.069	-	-	0.063	-	-
	Frozen LLM w/ PT	PT w/ KG-Text	0.078	0.191	0.308	0.051	0.152	0.250	0.074	0.165	0.245
		GraphToken w/ RAG (Perozzi et al., 2024)	0.268	0.421	0.466	0.186	0.433	0.576	0.326	0.515	0.624
		G-retriever (He et al., 2024)	0.274	0.532	0.650	0.342	0.619	0.739	0.275	0.487	0.612
		K-RagRec	0.435	0.725	0.831	0.600	0.850	0.913	0.508	0.690	0.780
		Improvement	58.6%	33.0%	27.8%	75.4%	37.3%	23.5%	55.8%	34.0%	25.0 %
	Fine-tuning	Lora w/ KG-Text	0.402	0.718	0.833	0.609	0.848	0.905	0.446	0.648	0.758
		Lora w/ K-RagRec	0.466	0.770	0.863	0.637	0.872	0.927	0.516	0.720	0.799
		Improvement	15.9%	7.2%	3.6%	4.5%	2.7%	2.4%	15.7%	11.1%	5.4%
LLama-3	**Inference-only**	KG-Text (Wu et al., 2023b)	0.095	-	-	0.060	-	-	0.054	-	-
		KAPING (Baek et al., 2023)	0.084	-	-	0.069	-	-	0.062	-	-
	Frozen LLM w/ PT	PT w/ KG-Text	0.134	0.294	0.433	0.094	0.205	0.296	0.083	0.207	0.314
		GraphToken w/ RAG (Perozzi et al., 2024)	0.355	0.622	0.737	0.473	0.719	0.805	0.428	0.567	0.661
		G-retriever (He et al., 2024)	0.352	0.632	0.746	0.502	0.736	0.796	0.417	0.584	0.682
		K-RagRec	0.472	0.704	0.765	0.634	0.779	0.818	0.514	0.662	0.723
		Improvement	32.9%	11.4%	2.5%	26.3%	5.8%	1.6%	20.0%	13.4%	6.0%
	Fine-tuning	Lora w/ KG-Text	0.449	0.694	0.750	0.648	0.757	0.790	0.490	0.638	0.698
		Lora w/ K-RagRec	0.498	0.712	0.771	0.674	0.786	0.817	0.546	0.672	0.733
		Improvement	10.9%	2.6%	2.8%	4.0%	3.8%	3.4%	11.4%	5.3%	5.0%
QWEN2	**Inference-only**	KG-Text (Wu et al., 2023b)	0.160	-	-	0.174	-	-	0.194	-	-
		KAPING (Baek et al., 2023)	0.196	-	-	0.208	-	-	0.220	-	-
	Frozen LLM w/ PT	PT w/ KG-Text	0.190	0.371	0.499	0.259	0.397	0.494	0.303	0.451	0.553
		GraphToken w/ RAG (Perozzi et al., 2024)	0.259	0.487	0.608	0.370	0.550	0.632	0.365	0.568	0.658
		G-retriever (He et al., 2024)	0.304	0.551	0.644	0.389	0.606	0.685	0.355	0.552	0.658
		K-RagRec	0.416	0.712	0.829	0.586	0.842	0.904	0.502	0.686	0.767
		Improvement	36.8%	29.2%	28.4%	50.6%	38.9%	32.0%	37.5%	20.8%	16.6 %
	Fine-tuning	Lora w/ KG-Text	0.400	0.701	0.815	0.601	0.842	0.906	0.478	0.667	0.751
		Lora w/ K-RagRec	0.466	0.763	0.860	0.631	0.868	0.928	0.510	0.704	0.780
		Improvement	16.5%	8.8%	5.5%	5.0%	3.1 %	2.4%	6.7%	5.5%	3.9%

source backbone LLMs: LLama-2-7b (Touvron et al., 2023), LLama-3-8b (Dubey et al., 2024), and QWEN2-7b (Yang et al., 2024). We present the results in Table 1. From the comparison, we have the following main observations:

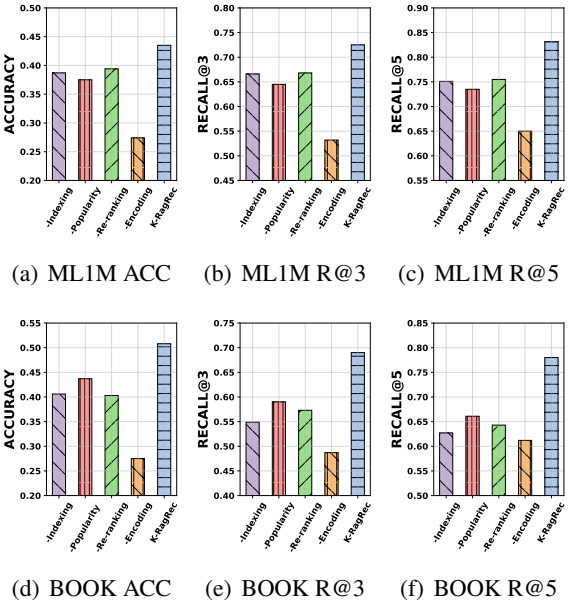
- Naively retrieve KG and augment LLM with text methods (i.e., KG-Text and KAPING), have limited recommendation accuracy on the MovieLens-1M and MovieLens-20M and Amazon Book datasets.
- Compared to other prompt tuning RAG methods, K-RagRec with LLama-2-7B as the backbone LLM leads to an average of 41.6% improvement over the sub-optimal baseline across all datasets. With LLama-3-8B and QWEN2-7B as the backbone LLM, K-RagRec also brought an average of 13% to 32% improvement, highlighting the effectiveness of our proposed method in augmenting LLMs' recommendation performance.
- Compared to the LoRA fine-tuning with the naive RAG approach, K-RagRec with prompt tuning achieves close to or better performance in most settings. Notably, K-RagRec achieves the best performance when fine-tuned with LoRA.

4.3 Ablation Study

To evaluate the impact of each component in our proposed framework, we conduct the ablation study to compare the K-RagRec with four ablated variants on MovieLens and Amazon Book datasets, using LLama-2-7B as the backbone LLM. Details of each ablation variant are provided in Appendix A.6. The results are illustrated in Figure 3. Observing the experiment results, we find that eliminating any component of the framework leads to a decrease in the overall performance of the recommendations, demonstrating the effectiveness of each module. Secondly, removing the GNN Encoder leads to a 37% decrease and a 45.9% decrease in the accuracy of the model on MovieLens and Amazon Book datasets, respectively, highlighting the significance of employing GNN to encode the structure of knowledge sub-graphs. Refer to Appendix A.6 for more details.

4.4 Efficiency Evaluation

In this sub-section, we evaluate the inference efficiency of our proposed K-RagRec framework com-



(a) ML1M ACC (b) ML1M R@3 (c) ML1M R@5

(d) BOOK ACC (e) BOOK R@3 (f) BOOK R@5

Figure 3: Comparison among K-RagRec and its four ablated variants on MovieLens-1M and Amazon Book datasets and LLama-2-7b across metrics Accuracy, Recall@3 and Recall@5.

Table 2: Comparison of the inference efficiency on the MovieLens-1M dataset and LLama-2-7b in seconds (s). ACC denote Accuracy.

Methods	ACC	Time (s)
w/o RAG	-	0.92
KG-Text	0.076	2.19
KAPING	0.079	6.47
GraphToken w/ RAG	0.268	3.14
G-retriever	0.274	5.86
K-RagRec	0.429	1.06

pared with baselines on the MovieLens-1M dataset and LLama-2-7b. We record the time cost for one inference utilizing two NVIDIA A6000-48G GPUs. The time cost for a single inference is reported in Table 2. By observing the experimental results, we notice that various KG RAG approaches significantly increase the inference time, which is due to the large scale of the KG. In contrast, K-RagRec achieves the best computational efficiency compared to various KG RAG methods and is only about 0.1s slower than direct inference without retrieval. These findings highlight the efficiency of K-RagRec and validate the effectiveness of our popularity selective retrieval policy.

4.5 Parameter Analysis

In this section, we evaluate the impact of three main hyper parameters of K-RagRec, namely popularity selective retrieval policy threshold p, retrieved knowledge sub-graph numbers K, and re-ranking knowledge sub-graph numbers N. In addition, we analyze the impact of various GNN encoder variants and different GNN layer numbers for the proposed framework in Appendix A.10 and A.11.

1) Impact of popularity selective retrieval policy threshold p: To understand how the popularity selective retrieval policy threshold p affects K-RagRec, we conduct experiments on MovieLens-1M and LLama-2-7b across two metrics. Results are shown in Figure 4. As the threshold p increases, the recommendation performance initially improves and then decreases. When the threshold p is set to a small value, only a few items are augmented. This leads to insufficient retrieval and poor recommendation accuracy. When p is set to a larger value, more items are retrieved for augmentation. However, due to the re-ranking sub-graph numbers N being fixed, some retrieved cold-start item knowledge sub-graphs are discarded or ranked at the back of the list, resulting in sub-optimal recommendation performance. On the other hand, as observed in Figure 4 (c), the inference time increases almost linearly with threshold p. Therefore, selecting an appropriate threshold p is crucial to balance the performance and inference time.

2) Impact of retrieved knowledge sub-graph numbers K and re-ranking sub-graph numbers N: In this part, we analyze the impact of two key hyper parameters, which are retrieved knowledge sub-graph numbers K and re-ranking sub-graph numbers N. First, to measure the impact of K, we perform experiments on the MovieLens dataset and fix $p = 50\%$, $N = 5$. As shown in Figure 5, some relevant knowledge sub-graphs may be overlooked when $K = 1$. On the other hand, larger values of K can introduce irrelevant information. Therefore, we set K equal to 3 in our experiments. Next, we evaluate the effect of N by fixing $p = 50\%$ and $K = 3$, and present the results in Figure 6. We observe that setting N to between 5 and 7 results in improved performance on the Amazon Book dataset. In general, it is important to carefully choose K and N based on the scale of the dataset and the KG.

5 Conclusion

In this paper, we propose a novel framework **K-RagRec** to augment the recommendation capability of LLMs by retrieving reliable and up-to-date knowledge from KGs. Specifically, we first introduce a GNN and PLM to perform semantic indexing of KGs, enabling both coarse and fine-grained

retrieval for KGs. To further improve retrieval efficiency, we introduce a popularity selective retrieval policy that determines whether an item needs to be retrieved based on its popularity. Notably, K-RagRec performs more expressive graph encoding of the retrieved knowledge sub-graphs, facilitating the LLM to effectively leverage the structure information and avoid long context input. Extensive experiments conducted on three real-world datasets demonstrate the effectiveness of our proposed framework.

6 Limitations

To the best of our knowledge, this work is a pioneering study in investigating knowledge-graph RAG for LLM-based recommendations. Therefore, we realise that this work still has the following three main limitations that can be improved in the future research:

- Firstly, due to the GPU resource constraints, we were only able to evaluate our framework on 7b and 8b models. Therefore, in future work, we plan to extend our method to larger models to fully assess its effectiveness and scalability.
- Additionally, we only utilize Freebase as the external KG as it is most commonly used for recommendation tasks. Thus we also aim to adopt other KGs, such as YAGO, DBpedia, and Wikipedia, to better understand how different knowledge sources impact the performance of the proposed method.
- Lastly, designing an intelligent selective retrieval policy for LLM-based recommender systems is an important and challenging task. In this work, we propose to leverage popularity to determine which items to retrieve to improve retrieval efficiency. In the future, we will investigate more flexible mechanisms (e.g., reinforcement learning) to dynamically update policies according to changes in user interest.

7 Acknowledgements

The research described in this paper has been partly supported by General Research Funds from the Hong Kong Research Grants Council (project no. PolyU 15207322, 15200023, 15206024, and 15224524), internal research funds from The Hong Kong Polytechnic University (project no. P0042693, P0048625, P0051361, P0052406, and P0052986).

References

Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling popularity bias in learning-to-rank recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*, pages 42–46.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Yihao Ang, Yifan Bao, Qiang Huang, Anthony KH Tung, and Zhiyong Huang. 2024. Tsgassist: An interactive assistant harnessing llms and rag for time series generation recommendations and benchmarking. *Proceedings of the VLDB Endowment*, 17(12):4309–4312.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*.

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1007–1014.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Òscar Celma and Perfecto Herrera. 2008. A new approach to evaluating novel recommendations. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 179–186.

Dario Di Palma. 2023. Retrieval-augmented recommender system: Enhancing recommender systems with large language models. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1369–1373.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024a. A survey on rag meeting llms: Towards

retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501.

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *The world wide web conference*, pages 417–426.

Wenqi Fan, Shijie Wang, Jiani Huang, Zhikai Chen, Yu Song, Wenzhuo Tang, Haitao Mao, Hui Liu, Xiaorui Liu, Dawei Yin, et al. 2024b. Graph machine learning in the era of large language models (llms). *arXiv preprint arXiv:2404.14928*.

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

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 299–315.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

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

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*, pages 639–648.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630*.

Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 585–593.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Krishnaram Kenthapadi, Benjamin Le, and Ganesh Venkataraman. 2017. Personalized job recommendation system at linkedin: Practical challenges and lessons learned. In *Proceedings of the eleventh ACM conference on recommender systems*, pages 346–347.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*.

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

Tiziano Labruna, Jon Ander Campos, and Gorka Azkune. 2024. When to retrieve: Teaching llms to utilize information retrieval effectively. *arXiv preprint arXiv:2404.19705*.

Jiatong Li, Yunqing Liu, Wenqi Fan, Xiao-Yong Wei, Hui Liu, Jiliang Tang, and Qing Li. 2024. Empowering molecule discovery for molecule-caption translation with large language models: A chatgpt perspective. *IEEE Transactions on Knowledge and Data Engineering*.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2023. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*.

Costas Mavromatis and George Karypis. 2024. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions. *arXiv preprint arXiv:2004.10645*.

Bryan Perozzi, Bahare Fatemi, Dustin Zelle, Anton Tsitsulin, Mehran Kazemi, Rami Al-Rfou, and Jonathan Halcrow. 2024. Let your graph do the talking: Encoding structured data for llms. *arXiv preprint arXiv:2402.05862*.

Haohao Qu, Wenqi Fan, Zihuai Zhao, and Qing Li. 2024. Tokenrec: Learning to tokenize id for llm-based generative recommendation. *arXiv preprint arXiv:2406.10450*.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Priyanka Sen, Sandeep Mavadia, and Amir Saffari. 2023. Knowledge graph-augmented language models for complex question answering. In *Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE)*, pages 1–8.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. 2020. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. 2017. Graph attention networks. *stat*, 1050(20):10–48550.

Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. 2023. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *arXiv preprint arXiv:2308.13259*.

Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 806–815.

Junda Wu, Cheng-Chun Chang, Tong Yu, Zhankui He, Jianing Wang, Yupeng Hou, and Julian McAuley. 2024. Coral: Collaborative retrieval-augmented large language models improve long-tail recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3391–3401.

Shengqiong Wu, Hao Fei, Leigang Qu, Wei Ji, and Tat-Seng Chua. 2023a. Next-gpt: Any-to-any multimodal llm. *arXiv preprint arXiv:2309.05519*.

Yike Wu, Nan Hu, Sheng Bi, Guilin Qi, Jie Ren, Anhuan Xie, and Wei Song. 2023b. Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering. *arXiv preprint arXiv:2309.11206*.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Junjie Zhang, Yupeng Hou, Ruobing Xie, Wenqi Sun, Julian McAuley, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2024. Agentcf: Collaborative learning with autonomous language agents for recommender systems. In *Proceedings of the ACM on Web Conference 2024*, pages 3679–3689.

Yang Zhang, Fuli Feng, Jizhi Zhang, Keqin Bao, Qifan Wang, and Xiangnan He. 2023. Collm: Integrating collaborative embeddings into large language models for recommendation. *arXiv preprint arXiv:2310.19488*.

Jujia Zhao, Wenjie Wang, Xinyu Lin, Leigang Qu, Jizhi Zhang, and Tat-Seng Chua. 2023. Popularity-aware distributionally robust optimization for recommendation system. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4967–4973.

Zihuai Zhao, Wenqi Fan, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Zhen Wen, Fei Wang, Xiangyu Zhao, Jiliang Tang, et al. 2024. Recommender systems in the era of large language models (llms). *IEEE Transactions on Knowledge and Data Engineering*.

A Appendix

A.1 Implementation Details

The hyper parameters used for K-RagRec and their corresponding values are shown in Table 4. The first part provides the general training setting for K-RagRec. The second part presents the details of ($\text{GNN}^{\text{Indexing}}$, $\text{GNN}^{\text{Encoding}}$). Then we list the Lora and acceleration settings. Lastly, we provide the hyper parameters for retrieval, including candidate item number M, popularity selective retrieval policy threshold p, retrieve knowledge sub-graphs numbers K, and re-ranking knowledge sub-graphs numbers N. If not specified, we run all methods three times with different random seeds and report the averaged results.

Table 3: Basic statistics of three datasets and the KG. "Items in KG" indicates the number of items that appeared in both the KG and the dataset.

Datasets	MovieLens-1M	MovieLens-20M	Amazon Book
User	6,038	138,287	6,106,019
Item	3,533	20,720	1,891,460
Interaction	575,281	9,995,410	13,886,788
Items in KG	3,498	20,139	91,700
Entitys	250,631	1,278,544	186,954
Relations	264	436	16
KG triples	348,979	1,827,361	259,861

A.2 Graph Neural Networks

GNNs are a critical technique in graph machine learning and are widely employed in various graph tasks. By iteratively updating node representations through aggregating information from neighboring nodes, GNNs effectively capture the underlying topology and relational structure of graphs. Formally, a typical GNN operation can be formulated as follows:

$$\mathbf{x}_j^{(l+1)} = \mathbf{x}_j^{(l)} \oplus \text{AGG}^{(l+1)}\left(\left\{\mathbf{x}_i^{(l)} \mid i \in \mathcal{N}_{x_j}\right\}\right), \tag{10}$$

where $\mathbf{x}_j^{(l+1)}$ express node j's feature on the l-th layer, and $\mathcal{N}(x_j)$ is the set of neighbours of node j. AGG is a aggregation function to aggregates neighbors' features, and $\oplus$ combines neighbors' information with the node itself.

A.3 More Related Work

The remarkable breakthroughs in LLMs have led to their widespread adoption across various fields, particularly in the recommendations. Given powerful reasoning and generalization capabilities, many studies have actively attempted to harness the power of the LLM to enhance recommender systems (Geng et al., 2022; Bao et al., 2023; Qu et al., 2024; Wei et al., 2024; Zhang et al., 2023). For example, P5 (Geng et al., 2022) proposes an LLM-based recommendation model by unifying pre-training, prompting, and prediction for various recommendation tasks, such as sequential recommendation and rating predictions. Furthermore, Tallrec (Bao et al., 2023) fine-tunes LLM (i.e., LLaMA-7B) to align with recommendation data for sequential recommendations. To further capture higher-order collaborative knowledge and enhance the model's ability to generalize users and items, TokenRec (Qu et al., 2024) proposes a masked vector-quantized tokenizer to tokenize users and items in LLM-based recommendations. Despite their effectiveness, these models often face challenges, such as hallucinations and the lack of up-to-date knowledge. While fine-tuning can partially mitigate these issues, it is resource-costly and time-consuming due to the massive parameters of LLMs. To overcome these challenges, we propose a knowledge retrieval augmented recommendation framework that leverages external KGs to provide reliable and up-to-date knowledge instead of costly fine-tuning.

Table 4: Statistics of Hyper Parameters.

Item	Value
batch size	5
epochs	3
grad steps	2
learning rate	1e-5
Indexing layer numbers	4
Indexing hidden dimension	1024
Encoding layer numbers	4
Encoding hidden dimension	1024
Encoding head numbers	4
lora_r	8
lora_alpha	16
lora_dropout	0.1
int8	True
fp16	True
candidate item numbers	20
threshold p	50%
top-K	3
top-N	5

A.4 Comparison with Existing Methods

Existing LLM-based recommender systems usually require frequent fine-tuning on specific datasets to address the lack of knowledge and hallucinations, which is time-consuming and costly. To solve these challenges and avoid costly fine-tuning, our work

is the first to augment the recommendation performance of LLMs by retrieving structured data (i.e., knowledge graph). Specifically, our approach differs from existing work in the following ways:

- K-RagRec introduces an indexing GNN to efficiently retrieve structured data to enhance the recommendation capability of LLMs. Although some GraphRAG approaches also introduce GNNs to capture higher-order neighbourhood information, they only apply the last layer of the GNN representation, leading to coarse retrieval (He et al., 2024; Mavromatis and Karypis, 2024). In contrast, our approach leverages the representation of each GNN layer to retrieve nuanced knowledge of both coarse and fine-grained graph structures from KG, achieving a more comprehensive and precise retrieval.
- In the recommendation domain that pursues inference speed, excessive retrieval time can seriously degrade the user experience resulting in user churn. Although many studies have explored selective retrieval for RAGs (Yan et al., 2024; Jiang et al., 2023), it is still an open question to determine whether an item needs to be retrieved and to reduce the retrieval time in the recommendation domain. We propose to use popularity to decide whether an item needs to be retrieved based on power law distribution, which greatly reduces the retrieval time. Table 2 shows that K-RagRec achieves inference times close to direct inference while maintaining high recommendation accuracy.
- Typical RAG methods usually incorporate the retrieved content into the prompt as text (Wu et al., 2023b; Baek et al., 2023). However, vanilla RAG methods rely on documents and paragraphs often introduce unnecessary noise and even harmful disturbance, which can negatively impact the accuracy and reliability of recommendations. In addition, the structural relationships between entities are overlooked in typical RAG, resulting in the sub-optimal reasoning capability of LLM-based recommender systems. We propose to incorporate the retrieved knowledge subgraphs (Knowledge-GraphRAG) into the query as a graph prompt, which facilitates LLMs to better understand the retrieved knowledge subgraphs and avoids long contexts.

A.5 Comparison with 10 Candidate Items

In this section, we conduct additional experiments to evaluate the effectiveness of K-RagRec with a candidate item number $M = 10$. In this setting, we randomly select nine negative samples with the target item. The results are presented in Table 5. We exclude the results of some backbone LLM models (e.g., LLama-3 and QWEN2), as similar observations as Table 1 can be found. Observing the experimental results, we can note that our proposed K-RagRec method consistently outperforms all baseline methods on MovieLens and Amazon Book datasets, further highlighting the effectiveness of our framework.

Table 5: Performance comparison of different KG RAG-enhanced LLM recommendations with candidate item numbers $M = 10$ on the MovieLens and Amazon Book dataset and LLama-2-7b across two metrics. The best performances are labeled in bold. ACC and R@3 denote Accuracy and Recall@3, respectively.

Methods	MovieLens-1M		Amazon Book	
	ACC	R@3	ACC	R@3
KG-Text	0.185	-	0.142	-
KAPING	0.165	-	0.119	-
PT w/ KG-Text	0.159	0.493	0.123	0.384
GraphToken w/ RAG	0.512	0.753	0.444	0.682
G-retriever	0.469	0.721	0.367	0.610
K-RagRec	**0.568**	**0.779**	**0.606**	**0.770**

A.6 Ablation Study Setting

To assess the impact of each module in K-RagRec, we compare the framework with four ablated variants: K-RagRec (-*Indexing*), K-RagRec (-*Popularity*), K-RagRec (-*Re-ranking*), and K-RagRec (-*Encoding*). (1) K-RagRec (-*Indexing*) eliminates the $\text{GNN}^{\text{Indexing}}$ and stores semantic information of PLM in the knowledge vector database. For the retrieved nodes, we extract their second-order sub-graphs as the retrieved knowledge sub-graphs. (2) K-RagRec (-*Popularity*) does not apply the popularity selective retrieval policy and retrieves all items from the user's historical interactions. (3) (-*Re-ranking*) removes the re-ranking module and inputs the knowledge sub-graphs directly. (4) K-RagRec (-*Encoding*) removes the $\text{GNN}^{\text{Encoding}}$ and replaces it with a trainable soft prompt. The retrieved knowledge sub-graphs will be added to the prompt as triples (e.g., *{Moonraker, film writer film, Christopher Wood (writer)}*).

A.7 Generalization Study

To evaluate the generalization capability of our proposed framework in the zero-shot setting, we trained a version of the model on the MovieLens-

Table 6: The generalization results for our K-RagRec model in a zero-shot setting. In this setting, our models are trained on MovieLens-1M dataset and evaluated on MovieLens-20M and Amazon Book datasets. ACC and R@k denote Accuracy and Recall@k, respectively.

Models	Methods	MovieLens-20M			Amazon Book		
		ACC	R@3	R@5	ACC	R@3	R@5
LLama2	K-RagRec	0.539	0.740	0.795	0.390	0.581	0.671
	Lora w/K-RagRec	0.539	0.783	0.863	0.405	0.580	0.796
LLama3	K-RagRec	0.597	0.797	0.839	0.428	0.628	0.706
	Lora w/K-RagRec	0.611	0.775	0.814	0.424	0.622	0.732
QWEN	K-RagRec	0.507	0.769	0.861	0.418	0.612	0.687
	Lora w/K-RagRec	0.545	0.814	0.897	0.441	0.623	0.706

Table 7: Quantitative comparison of hallucination on the MovieLens-1M dataset. Δ denotes the reduction in hallucinations for K-RagRec compared to Direct Inference.

Models	Direct Inference	K-RagRec	Δ
LLama-2	39.1%	2.7%	93.1%
QWEN2	4.7%	0.9%	80.9%

1M dataset and assessed it on the MovieLens-20M and Amazon Book datasets. The experiment results are shown in Table 6. We note that although the K-RagRec performance in the zero-shot setting is slightly degraded when compared to the well-trained model in Table 1, it still demonstrates 21.6% improvement over SOTA baselines on the MovieLens-20M dataset. Furthermore, despite the differences between the book recommendation and movie recommendation tasks, the model trained on MovieLens-1M delivers about 8.7% improvement in the zero-shot setting compared to prompt-tuned baselines on the Amazon Book dataset. The experimental results demonstrate that K-RagRec exhibits strong generalization capabilities and is adaptable across different domains.

A.8 Study of Hallucination

In this section, we present a qualitative analysis of hallucinations in the LLama-2-7b and QWEN2 models on the MovieLens dataset. Specifically, we include a few fictional movies in the candidate items to observe the probability of the fictional movie being recommended. We compare direct recommendations and recommendations augmented with K-RagRec, and the results are shown in Table 7. We note that K-RagRec significantly reduced hallucinations by 93.1% compared to direct inference on LLama-2. In contrast to LLama-2, QWEN2 rarely recommends fictional movies. Nevertheless, K-RagRec reduced hallucinations by 80.9%, demonstrating the effectiveness of our approach in addressing hallucinations.

Table 8: Performance comparison of different KG RAG-enhanced LLM recommendation methods on the cold-start dataset and QWEN2 across three metrics. The best performances are labeled in bold. ACC and R@k denote Accuracy and Recall@k, respectively.

Methods	ACC	R@3	R@5
PT w/ KG-Text	0.106	0.239	0.395
GraphToken w/ RAG	0.258	0.473	0.620
G-retriever	0.185	0.384	0.488
K-RagRec	**0.406**	**0.705**	**0.834**

Table 9: Comparison of different GNN Encoders on the MovieLens-1M dataset and LLama-2-7b across three metrics. We use bold fonts to label the best performance. ACC and R@k denote Accuracy and Recall@k, respectively.

GNN Types	ACC	R@3	R@5
GCN (Kipf and Welling, 2016)	0.397	0.704	0.809
GAT (Velickovic et al., 2017)	0.420	0.693	0.804
Graph Transformer (Shi et al., 2020)	**0.429**	**0.711**	0.779
GraphSAGE (Hamilton et al., 2017)	0.418	0.699	**0.823**

A.9 Study of Cold Start Recommendation

The cold start problem is an important issue in most recommendation research. To comprehensively evaluate our approach, we particularly design a case study to evaluate the model's recommendation performance under the cold-start setting. Specifically, we construct a separate cold-start dataset based on the MovieLens dataset that only contains these identified cold-start items as target items. We compare K-RagRec with three KG RAG-enhanced LLM recommendation methods, and the experiment results are shown in Table 8. The results demonstrate that our proposed K-RagRec still has satisfactory performance under the cold-start recommendation scenario, highlighting the effectiveness of our framework in all the cases.

A.10 Study of Four GNN Encoders

To further understand the generality of our proposed approach, we conduct a comparative study of four variants applying different GNN encoders. Specifically, we compare GCN (Kipf and Welling, 2016), GAT (Velickovic et al., 2017), Graph Transformer (Shi et al., 2020), and GraphSAGE (Hamilton et al., 2017) as four K-RagRec variants of GNN encoder. Specifically, Graph Convolutional Net-

Table 10: Comparison of different GNN layers on the Amazon Book dataset and LLama-2-7b across three metrics. ACC and R@k denote Accuracy and Recall@k, respectively.

GNN Layers	ACC	R@3	R@5
3 layers	0.496	0.653	0.736
4 layers	0.506	0.690	0.780
5 layers	0.498	0.656	0.729

work (GCN) (Kipf and Welling, 2016) first introduces convolutional operations to graph-structured data. By aggregating features from neighboring nodes, GCN facilitates the learning of rich node representations. GraphSAGE (Hamilton et al., 2017) learns an aggregation function that samples and combines features from a node's local neighborhood in an inductive setting, enabling the effective use of new nodes. Graph Attention Network (GAT) (Velickovic et al., 2017) further incorporates attention mechanisms, allowing the model to dynamically assign varying attention to neighboring nodes, thereby enhancing the focus on the most relevant information. Inspired by the success of the transformer, the Graph Transformer (Shi et al., 2020) adapts transformer architectures to graph data, enhancing the modeling of graphs, particularly textual graphs.

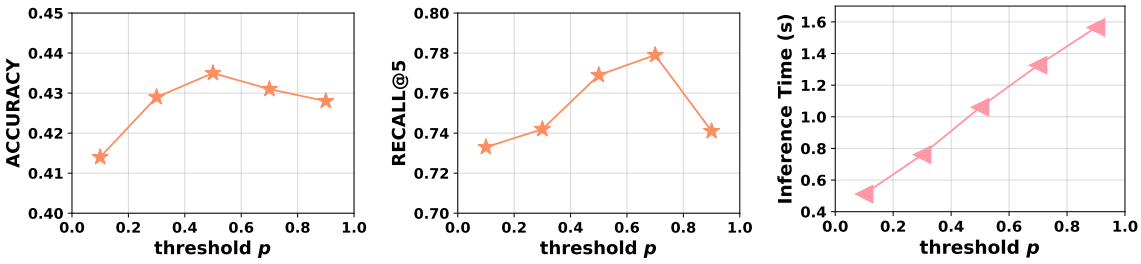
We report the experiment results on the MovieLens-1M dataset and the LLama-2-7b backbone in Table 9. It is noted that the GCN Encoder variant method performs second-best on the Recall@5 metric, although it is slightly worse than other GNN encoders on the Accuracy metric. Overall, four GNN encoder variants exhibit close performance on the MovieLens dataset, highlighting the generality and robustness of our framework across different GNN encoders.

A.11 Study of GNN Layer Numbers

In this subsection, we evaluate the impact of the number of GNN layers on model performance. We vary the numbers of the GNN layer numbers in the range of {3, 4, 5} and test on the Amazon Book dataset and LLama-2-7b across three metrics. As observed in Table 10, the model performance first improves and then decreases as the number of GNN layers increases, and the model achieves the best results across the three metrics when setting the number of GNN layers is set to four. Therefore, a smaller number of GNN layers may not have sufficient depth to capture the intricate relationships and dependencies in the graph, leading to sub-optimal performance. On the other hand, too many layers can result in indistinguishable node representations. Thus, selecting the optimal number of GNN layers is crucial for effective model training.

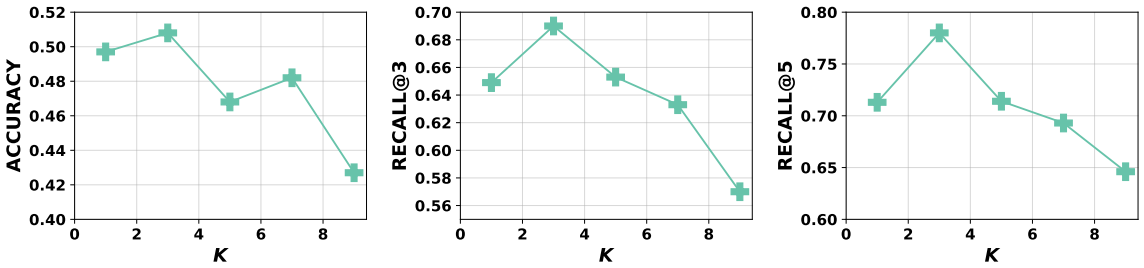
A.12 Used Prompt

In this part, we present the prompts designed for movie recommendations and book recommendations. We show two examples in Table 11, and set the candidate items M equal to 20. For inference, we leverage the model to make the prediction based on the user's recent watching history and candidate items.



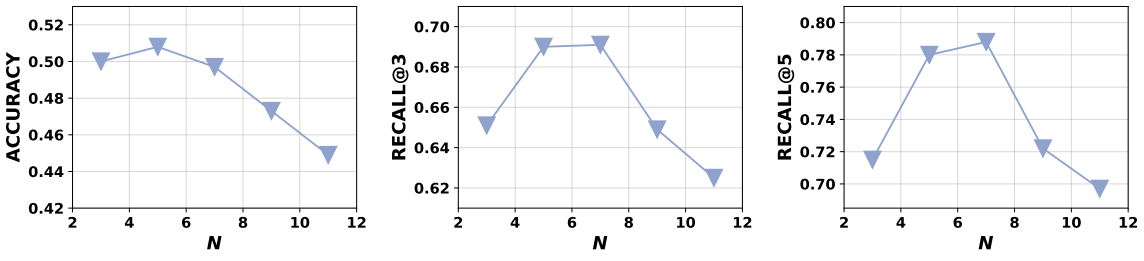
(a) Accuracy (b) Recall@5 (c) Inference Time

Figure 4: Effect of popularity selective retrieval policy threshold p on MovieLens-1M and LLama-2-7b across metrics Accuracy, Recall@5 and inference time (seconds) for K-RagRec.



(a) Amazon Book: Accuracy (b) Amazon Book: Recall@3 (c) Amazon Book: Recall@5

Figure 5: Effect of retrieved knowledge sub-graph numbers K on Amazon Book datasets and LLama-2-7b across metrics Accuracy, Recall@3 and Recall@5 for K-RagRec.



(a) Amazon Book: Accuracy (b) Amazon Book: Recall@3 (c) Amazon Book: Recall@5

Figure 6: Effect of re-ranking knowledge sub-graph numbers N on Amazon Book datasets and LLama-2-7b across metrics Accuracy, Recall@3 and Recall@5 for K-RagRec.

Table 11: Example of the used prompt for K-RagRec. The user's recent watching/reading history and candidate items are marked in red and blue, respectively.

Datasets	Used Prompt
Movies	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. Instruction: Given the user's watching history, select a film that is most likely to interest the user from the options. Watching history: {"History of the World: Part I", "Romancing the Stone", "Fast Times at Ridgemont High", "Good Morning, Vietnam", "Working Girl", "Cocoon", "Splash", "Pretty in Pink", "Terms of Endearment", "Bull Durham"}. Options: {A: "Whole Nine Yards", B: "Hearts and Minds", C: "League of Their Own", D: "Raising Arizona", E: "Happy Gilmore", F: "Brokedown Palace", G: "Man Who Knew Too Much", H: "Light of Day", I: "Tin Drum", J: "Blair Witch Project", K: "Red Sorghum", L: "Flintstones in Viva Rock Vegas", M: "Anna, N: Roger & Me" O: "Land and Freedom", P: "In Love and War", Q: "Go West", R: "Kazaam", S: "Thieves", T: "Friends & Lovers"}. Select a movie from options A to T that the user is most likely to be interested in.
Books	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. Instruction: Given the user's reading history, select a book that is most likely to interest the user from the options. Watching history: {"Practice Makes Perfect: Spanish Verb Tenses", "NTC's Dictionary of Common Mistakes in Spanish", "Streetwise Spanish Dictionary/Thesaurus", "Buscalo! (Look It Up!) : A Quick Reference Guide to Spanish Grammar and Usage", "La lengua que heredamos: Curso de espaol para bilinges", "Vox Diccionario De Sinonimos Y Antonimos", "Schaum's Outline of Spanish Vocabulary", "Nos Comunicamos (Spanish Edition)", "The Oxford Spanish Business Dictionary", "Bilingual Dictionary of Latin American Spanish"}. Options: {"A: Folk and Fairy Tales, Childcraft (Volume 3)", B: "The Waves", C: "Dead End Kids: Gang Girls and the Boys They Know", D: "Opera Stars in the Sun: Intimate Glimpes of Metropolitan Opera Personalities", E: "Five-Minute Erotica", F: "Spanish Verbs: Oxford Minireference", G: "Motorcycle Maintenance Techbook: Servicing & Minor Repairs for All Motorcycles & Scooters", H: "Father and Son: A Study of Two Temperaments (Classic, 20th-Century, Penguin)", I: "Manga Mania Fantasy Worlds: How to Draw the Amazing Worlds of Japanese Comics", J: "MCSE Designing a Windows Server 2003 Active Directory & Network Infrastructure: Exam 70-297 Study Guide and DVD Training System", K: "The Atmospheric Boundary Layer (Cambridge Atmospheric and Space Science Series)", L: "St. Augustine and St. Johns County: A pictorial history", M: "A Will to Survive: Indigenous Essays on the Politics of Culture, Language, and Identity", N: "Saved: A Guide to Success With Your Shelter Dog", O: "Mosaic (Star Trek Voyager)", P: "Fantastical Tarot: 78-Card Deck", Q: "American Sign Language-A Look at Its History, Structure and Community", R: "Warrior's Heart (Zebra Historical Romance)", S: "The New Money Management: A Framework for Asset Allocation", T: "Megabrain"}. Select a book from options A to T that the user is most likely to be interested in.