EditKG: Editing Knowledge Graph for Recommendation

Gu Tang
Shanghai Jiao Tong University
Shanghai, China
gutang@sjtu.edu.cn

Xiaoying Gan*
Shanghai Jiao Tong University
Shanghai, China
ganxiaoying@sjtu.edu.cn

Jinghe Wang
Shanghai Jiao Tong University
Shanghai, China
whalien-56@sjtu.edu.cn

Bin Lu
Shanghai Jiao Tong University
Shanghai, China
robinlu1209@sjtu.edu.cn

Lyuwen Wu
Shanghai Jiao Tong University
Shanghai, China
wlw2016@sjtu.edu.cn

Luoyi Fu
Shanghai Jiao Tong University
Shanghai, China
yiluofu@sjtu.edu.cn

Chenghu Zhou
Chinese Academy of Sciences
Beijing, China
zhouch@lreis.ac.cn

ABSTRACT

With the enrichment of user-item interactions, Graph Neural Networks (GNNs) are widely used in recommender systems to alleviate information overload. Nevertheless, they still suffer from the cold-start issue. Knowledge Graphs (KGs), providing external information, have been extensively applied in GNN-based methods to mitigate this issue. However, current KG-aware recommendation methods suffer from the *knowledge imbalance problem* caused by incompleteness of existing KGs. This imbalance is reflected by the long-tail phenomenon of item attributes, i.e., unpopular items usually lack more attributes compared to popular items. To tackle this problem, we propose a novel framework called *EditKG: Editing Knowledge Graph for Recommendation*, to balance attribute distribution of items via editing KGs. EditKG consists of two key designs: *Knowledge Generator* and *Knowledge Deleter*. Knowledge Generator generates attributes for items by exploring their mutual information correlations and semantic correlations. Knowledge Deleter removes the task-irrelevant item attributes according to the parameterized task relevance score, while dropping the spurious item attributes through aligning the attribute scores. Extensive experiments on three benchmark datasets demonstrate that EditKG significantly outperforms state-of-the-art methods, and achieves 8.98% average improvement. The implementations are available at https://github.com/gutang-97/2024SIGIR-EditKG.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

*Corresponding author.

SIGIR '24, July 14–18, 2024, Washington, DC, USA.

ACM ISBN 979-8-4007-0431-4/24/07
https://doi.org/10.1145/3626772.3657723

KEYWORDS

Recommendation, Knowledge Graph, Graph Neural Network, Knowledge Imbalance

ACM Reference Format:
Gu Tang, Xiaoying Gan, Jinghe Wang, Bin Lu, Lyuwen Wu, Luoyi Fu, and Chenghu Zhou. 2024. EditKG: Editing Knowledge Graph for Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3626772.3657723

1 INTRODUCTION

In the era of information explosion, recommender systems have become a critical tool to address information overload. They have been widely used in many online services, such as e-commerce, video streaming platforms, and online advertising. Meanwhile, with the continuous increase in online users and items, the interactions between them are becoming progressively complex. Therefore, Graph neural networks (GNNs) [26], which specialize in modeling such complex interaction relationships, have been extensively applied in recommender systems. For instance, models like Light-GCN [11] and NGCF [35] apply GNNs to the user-item interactions to learn more powerful item and user representations.

Although these GNN-based methods have demonstrated promising results, they still suffer from the cold-start problem [54] caused by the sparsity of user-item interactions. Therefore, many advanced strategies, e.g., meta-learning, contrastive learning, and knowledge graphs (KGs), are combined to mitigate this issue. Specifically, meta-learning based approaches [14, 17] enhance model adaptability to cold-start users and items via cross-task knowledge transfer. Contrastive learning based efforts [7, 8] incorporate the strengths of self-supervised learning to further enhance the representations of cold-start users and items. These methods incorporate advanced techniques to explore features of raw data. Differently, knowledge graphs (KGs) provide external information for items and alleviate the cold-start issue from a data perspective. Hence, KGs have been extensively applied to GNN-based recommendation methods, such as KGCN [33], KGAT [34], and KGCL [45].

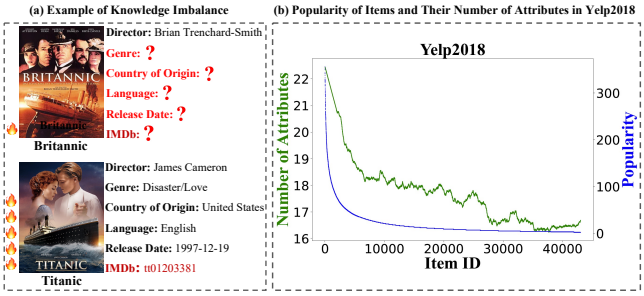

Figure 1: (a) Example of knowledge imbalance problem in KG-aware movie recommendation. (b) Popularity of Items and their number of attributes in Yelp2018.

Challenges. Despite effectiveness, current KG-aware recommendation methods still suffer from the *knowledge imbalance problem*. To be specific, KGs generally exhibit a significant incompleteness, while this incompleteness is more pronounced for unpopular items. That is to say, unpopular items usually lack more attributes compared to popular ones, leading to the Matthew Effect[1] in item attributes. We refer to this effect as the knowledge imbalance problem. From a statistical perspective, the knowledge imbalance problem is manifested as a long-tail phenomenon in item attributes, which results in an insufficient representation of unpopular items.

We illustrate the knowledge imbalance problem via a KG-aware movie recommendation example in Figure 1 (a). Compared to the popular film *Titanic*, the unpopular yet similar film *Britannic* lacks substantial attributes that should be present, such as "Genre", "Country of Origin", and "Language". This shows the Matthew Effect in item attributes. Additionally, we conduct a statistical analysis based on the real-world dataset Yelp2018 and its associated KG, depicted in Figure 1 (b). It demonstrates the relationship between the popularity of items and the corresponding number of attributes. The "popularity" is reflected by the interaction times of items. The "number of attributes" refers to the count of one-hop attributes for items in KG after smoothing. Obviously, item popularity exhibits a long-tail phenomenon, while the corresponding number of attributes demonstrates a similar pattern. This reveals the knowledge imbalance problem from a statistical perspective.

Therefore, the key to mitigating the knowledge imbalance problem lies in supplementing the lacking attributes for items. Nevertheless, achieving this goal encounters two major challenges: (i) *How to acquire potential item attributes?* In the field of knowledge graph completion [13], the process of attribute supplement typically involves identifying the lacking attributes of entities from their potential attributes [29, 42]. However, acquiring the potential attributes heavily relies on domain-specific expertise, which poses a challenge for automated acquisition. (ii) *How to drop the spurious and task-irrelevant item attributes?* Even after successfully acquiring the potential attributes, they often include both task-irrelevant and spurious attributes. Meanwhile, the existing KGs usually contain the task-irrelevant attributes [45, 56], such as the "IMDb: tt01203381" of film *Titanic* in Figure 1 (a). These negative attributes result in an adverse shift in item representation. Furthermore, due to the lack of explicit features in these negative attributes, recognizing and dropping them becomes a challenge.

Contributions. To tackle the aforementioned challenges, we propose a novel framework called EditKG: Editing Knowledge Graph for Recommendation, which solves the knowledge imbalance problem via editing KG. Concretely, EditKG solves the aforementioned challenge (i) by proposing a **Knowledge Generator**. This module explores the attribute correlations between items from both mutual information and semantic perspectives. These correlations are integrated to guide the attribute transfer, thereby generating potential attributes. Thereafter, to address the above-mentioned challenge (ii), we propose a **Knowledge Deleter**. This module learns the parameterized task relevance score to depict the characteristic of task-irrelevant attributes, thereby dropping them effectively. Moreover, inspired by transfer learning [16, 41], we propose an Expertise Transfer Mechanism. It iteratively trains a knowledge graph completion (KGC) model, and considers the item attribute scores generated by both the KGC model and the Knowledge Deleter as their expertise feature. By aligning these expertise features, this mechanism enables Knowledge Deleter to effectively recognize spurious attributes. Furthermore, to avoid gradient conflicts [46] during the expertise feature alignment, we employ a gradient match strategy, which stabilizes the training process of EditKG and improves its performance.

In summary, our key contributions can be outlined as follows:

- We explore the knowledge imbalance problem in KG-aware recommendation. As far as we know, this is the first work to study this challenging problem. To alleviate this issue, we propose a novel framework EditKG.
- The Knowledge Generator enables EditKG to automatically generate potential attributes for items by exploring their mutual information correlations and semantic correlations.
- The proposed Knowledge Deleter learns the parameterized task relevance score to drop task-irrelevant attributes. Furthermore, this module incorporates the expertise of KGC model to drop spurious attributes.

We conduct comprehensive experiments on three benchmark datasets. Results demonstrate that EditKG consistently outperforms state-of-the-art KG-aware recommendation methods, and achieves 8.98% average improvement.

2 PROBLEM FORMULATION

This section starts with an explanation of the symbolic concepts relevant to this paper, subsequently presenting a formal definition of the KG-aware recommendation task.

User-Item interaction Graph. In this paper, we view implicit feedback (e.g., click or purchase items) of users as a bipartite graph $\mathcal{G} = \{(u_i, y_{u_i,v_i}, v_i) | u_i \in \mathcal{U}, v_i \in \mathcal{V}\}$, where $\mathcal{U}$ is user set and $\mathcal{V}$ is item set. The binary value $y_{u_i,v_i} = 1$ if user u_i interacted with item v_i, and vice versa $y_{u_i,v_i} = 0$.

Knowledge Graphs. Knowledge graphs (KGs) store a wealth of item-related attributes in the form of triplets. We formally define this structured representation as a heterogeneous graph $\mathcal{G}_k = \{(h, r, t) | h, t \in \mathcal{E}, r \in \mathcal{R}\}$, where $\mathcal{E}$ and $\mathcal{R}$ are entity set and relation set of KGs, respectively. To be more specific, head entity h and tail entity t are associated with relation r in each triplet (h, r, t). For

[1]The rich get richer and the poor get poorer

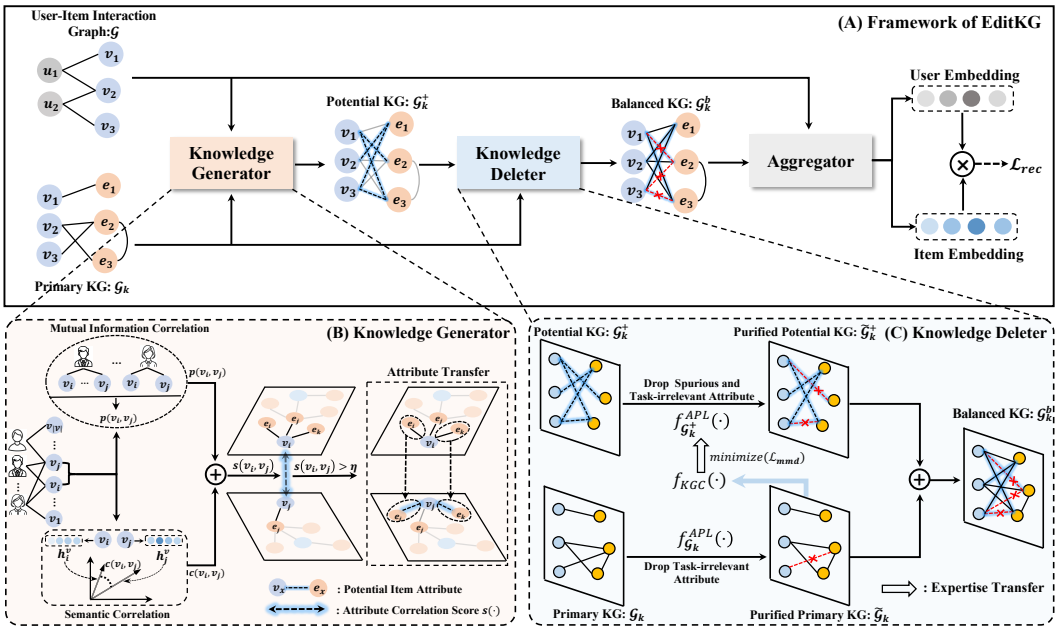

Figure 2: The overall architecture of the proposed EditKG. (A) demonstrates the framework of EditKG, (B) and (C) respectively detail the structural designs of two critical modules, i.e., Knowledge Generator and Knowledge Deleter.

instance, *(Titanic, Movie.Genre, Love)*, which characterizes the fact that the genre of the movie '*Titanic*' is '*Love*'. It is worth noting that the item set is a subset to the entity set, i.e., $\mathcal{V} \subset \mathcal{E}$. To facilitate understanding, we refer to $\mathcal{G}_k$ as the primary KG in our paper.

Task Formulation. We formulate the KG-aware recommendation as follows: based on the user-item interaction graph $\mathcal{G}$ and the primary KG $\mathcal{G}_k$, our goal is to train a model $\mathcal{F}(u_i, v_i|\mathcal{G}, \mathcal{G}_k, \Theta)$ to predict the probability of user u_i to adopt item v_i, where Θ represents trainable parameters of model $\mathcal{F}$.

3 METHODOLOGY

3.1 Knowledge Generator

In this subsection, we aim to generate the potential item attributes. As previously mentioned, acquiring the potential attributes poses a challenge, as it heavily relies on domain-specific prior knowledge [22].

Fortunately, our findings reveal that user-item interactions embody substantial prior knowledge: the attribute correlation between items is effectively delineated through the behavior pattern of users. This insight provides a viable and efficient approach to automatically generate the potential attributes.

3.1.1 ***Attribute Correlation Calculation.*** Our intuition suggests that if a strong attribute correlation exists between items, their attributes, obtained via KG, can potentially complement each other. Furthermore, the attribute correlation between items can be captured through user-item interactions. Specifically, inspired by normalized point mutual information (NPMI) [5], we calculate the mutual information correlation between items based on the interaction data of all users. It reflects the attribute correlation between items from the mutual information perspective. This procedure is defined as follows:

$$p(v_i, v_j) = -\frac{1}{\log C(v_i, v_j)} \log \frac{C(v_i, v_j)}{C(v_i)C(v_j)}, \tag{1}$$

where $C(v_i, v_j)$ denotes the co-occurrence frequency of item v_i and v_j within interaction data of all users, $C(v_i)$ denotes the appearance frequency of item v_i in the same data. $\log \frac{C(v_i,v_j)}{C(v_i)C(v_j)}$ is the point mutual information (PMI) [5], and $-\frac{1}{\log C(v_i,v_j)}$ normalizes its range to $[-1, 1]$. A higher value of $p(v_i, v_j)$ signifies a stronger attribute correlation between item v_i and v_j.

NPMI statistically quantifies the strength of attribute correlation between items from the mutual information perspective. To more accurately generate potential attributes for items, we additionally consider the semantic correlation $c(v_i, v_j)$ between them:

$$s(v_i, v_j) = p(v_i, v_j) + c(v_i, v_j), \tag{2}$$

$$c(v_i, v_j) = \frac{\mathbf{h}_i^v \cdot \mathbf{h}_j^v}{||\mathbf{h}_i^v|| \cdot ||\mathbf{h}_j^v||}, \tag{3}$$

where $\mathbf{h}_i^v, \mathbf{h}_j^v \in \mathbb{R}^d$ are the embeddings of item v_i and v_j, respectively. Based on cosine similarity, $c(v_i, v_j)$ assesses the attribute correlation between item v_i and v_j in the semantic space. It dynamically updates as the training progresses. The range of $c(v_i, v_j)$, same as $p(v_i, v_j)$, is $[-1, 1]$. It incorporates $p(v_i, v_j)$ to form a more credible attribute correlation score $s(v_i, v_j)$.

3.1.2 ***Attribute Transfer.*** For item v_i, we generate its potential attributes according to the attribute correlation score $s(v_i, v_j)$:

$$\widetilde{\mathcal{K}}_i = \begin{cases} \{(r, t)|(r, t) \in \mathcal{K}_j - (\mathcal{K}_j \cap \mathcal{K}_i)\} & if: s(v_i, v_j) \geq \eta, \\ \emptyset & if: s(v_i, v_j) < \eta, \end{cases} \tag{4}$$

where $\cap$ signifies the intersection, and η is a hyperparameter. $\mathcal{K}_i = \{(r,t)|(v_i,r,t) \in \mathcal{G}_k\}$ denotes the attribute set of item v_i in the primary KG $\mathcal{G}_k$. $\widetilde{\mathcal{K}}_i$ is the potential attribute set of item v_i. Importantly, if a strong attribute correlation exists between item v_i and v_j, a high-quality potential attribute (r,t) of item v_i should not only be driven from the attribute set $\mathcal{K}_j$ but also should be a novel attribute for item v_i, i.e., $(r,t) \in \mathcal{K}_j - (\mathcal{K}_j \cap \mathcal{K}_i)$. For clarity, we concatenate the potential attribute sets of all items to form a unified potential attribute set $\widetilde{\mathcal{K}}$. The potential KG $\mathcal{G}_k^+ = \{(h,r,t)|(r,t) \in \widetilde{\mathcal{K}}, h \in \mathcal{V}\}$ is generated by integrating all attributes in $\widetilde{\mathcal{K}}$ with their corresponding items.

3.2 Knowledge Deleter

The potential KG $\mathcal{G}_k^+$ contains a wealth of valuable item attributes, but also includes some noise. This occurs because the Knowledge Generator is capable only of generating potential attributes, where "potential" implies that these attributes may contain some noise, i.e., task-irrelevant and spurious attributes. Meanwhile, the primary KG $\mathcal{G}_k$ does not contain spurious attributes, as its construction relies on human prior knowledge. However, it often contains task-irrelevant attributes [45, 56]. This is due to the fact that recommender systems typically operate in highly specialized scenarios, whereas the primary KG $\mathcal{G}_k$ is designed for general scenarios.

Knowledge Deleter drops the above mentioned negative attributes in two steps: (i) Proposing an Attribute Purification Layer (APL) to drop the task-irrelevant attributes in the primary KG $\mathcal{G}_k$. (ii) Integrating the APL with an Expertise Transfer Mechanism to remove both task-irrelevant and spurious attributes in the potential KG $\mathcal{G}_k^+$.

3.2.1 ***Knowledge Deletion of Primary KG.*** We first discuss step (i), i.e., dropping the task-irrelevant attributes in the primary KG $\mathcal{G}_k$.

Attribute Purification Layer. Specifically, we introduce an Attribute Purification Layer (APL) to drop the task-irrelevant attributes in the primary KG. The APL is composed of a simple MLP, denoted as $f_{\mathcal{G}_k}^{APL}(\cdot)$. Meanwhile, the APL is supervised by a specific recommendation task, which enables APL to learn the task relevance score for each triplet of the primary KG $\mathcal{G}_k$:

$$\mathcal{P}_k = f_{\mathcal{G}_k}^{APL}(\mathcal{G}_k;\theta_{\mathcal{G}_k}) = [p_k^1,\cdots,p_k^{|\mathcal{G}_k|}] \tag{5}$$

$$p_k^i = f_{\mathcal{G}_k}^{APL}(\mathcal{G}_k^i;\theta_{\mathcal{G}_k}) = MLP([\mathbf{e}_h^i||\mathbf{e}_r^i||\mathbf{e}_t^i]), \tag{6}$$

where $||$ and σ denote the concatenate operation and the sigmoid function, respectively. $\theta_{\mathcal{G}_k}$ are the learnable parameters of $f_{\mathcal{G}_k}^{APL}(\cdot)$. $\mathbf{e}_h^i, \mathbf{e}_r^i, \mathbf{e}_t^i \in \mathbb{R}^d$ correspond to the embeddings of head entity h, relation r, and tail entity t in the i-th triplet of $\mathcal{G}_k$, respectively. A lower score p_k^i suggests that the corresponding triplet demonstrates a stronger task irrelevance. $\mathcal{P}_k \in \mathbb{R}^{|\mathcal{G}_k|}$ denotes the attribute score vector of $\mathcal{G}_k$ related to task relevance.

Then, we set the task relevance score p_k^i to 0 if $p_k^i \le \mu$, as its corresponding triplet $\mathcal{G}_k^i$ shows a significant task irrelevance. Meanwhile, we employ the Gumbel-Max reparameterization trick [18, 19] to ensure that this process differentiable:

$$\mathcal{G}_k^- = \{\mathcal{G}_k^i | m_k^i > 0, i \in \{1,\cdots,|\mathcal{G}_k|\}\}, \tag{7}$$

$$m_k^i = \mathbb{I}[\sigma(p_k^i + log(\epsilon) - log(1-\epsilon)) > \mu], \tag{8}$$

where $\epsilon \in \text{Uniform}(0,1)$ is a random variable, σ denotes the sigmoid function, and $\mu \in (0,1)$ is a hyperparameter. The $\mathbb{I}[\cdot]$ in Eq (8) is an indicator function, i.e., $\mathbb{I}[x > \mu] = x$ and $\mathbb{I}[x \le \mu] = 0$. $m_k^i = 0$ indicates that the triplet $\mathcal{G}_k^i$ is a task-irrelevant triplet. Based on the refined attribute score vector $\mathcal{M}_k = [m_k^1,\cdots,m_k^{|\mathcal{G}_k|}]$, we drop the task-irrelevant attributes in the primary KG $\mathcal{G}_k$, thereby forming an unbiased KG $\mathcal{G}_k^-$ through Eq (7). Simultaneously, for ease of representation, we integrate the primary KG $\mathcal{G}_k$ with its corresponding $\mathcal{M}_k$ to form a purified primary KG $\widetilde{\mathcal{G}}_k = \{(h,r,t,m^{(h,r,t)})|(h,r,t) \in \mathcal{G}_k, m^{(h,r,t)} \in \mathcal{M}_k\}$.

3.2.2 ***Knowledge Deletion of Potential KG.*** Then, we employ another APL to obtain attribute score vector for the potential KG $\mathcal{G}_k^+$, which is defined as follows:

$$\mathcal{P}_k^+ = f_{\mathcal{G}_k^+}^{APL}(\mathcal{G}_k^+;\theta_{\mathcal{G}_k^+}), \tag{9}$$

where $\theta_{\mathcal{G}_k^+}$ are the trainable parameters of $f_{\mathcal{G}_k^+}^{APL}(\cdot)$, $\mathcal{P}_k^+ \in \mathbb{R}^{|\mathcal{G}_k^+|}$ denotes the attribute score vector of potential KG $\mathcal{G}_k^+$ related to task relevance. However, the potential KG $\mathcal{G}_k^+$ contains both task-irrelevant and spurious attributes, while $f_{\mathcal{G}_k^+}^{APL}(\cdot)$ is unable to drop the spurious attributes.

Therefore, a natural idea is to incorporate a knowledge graph completion (KGC) model [4, 30] skilled in identifying spurious attributes to address this issue. Specifically, there are two intuitive strategies: (i) Sequential method: We employ a well-trained KGC model along with the $f_{\mathcal{G}_k^+}^{APL}(\cdot)$ to sequentially drop the spurious and task-irrelevant attributes in the potential KG $\mathcal{G}_k^+$. (ii) Joint method: We take the unbiased KG $\mathcal{G}_k^-$ as additional supervision data to construct an auxiliary task for $f_{\mathcal{G}_k^+}^{APL}(\cdot)$, thereby enabling $f_{\mathcal{G}_k^+}^{APL}(\cdot)$ to drop the spurious attributes. However, the sequential method suffers from the error accumulation [2]. Additionally, the joint method is constrained by the domain shift issue [51], since KG data and recommendation data originate from different domains and exhibit significant distributional differences.

Expertise Transfer Mechanism. Inspired by transfer learning [21, 50], we propose an Expertise Transfer Mechanism to refine the above mentioned joint method. This mechanism iteratively trains a knowledge graph completion (KGC) model based on the unbiased KG $\mathcal{G}_k^-$. Then, it considers the attribute scores generated by the KGC model and $f_{\mathcal{G}_k^+}^{APL}(\cdot)$ as their expertise features. By aligning these expertise features, this mechanism transfers the expertise of the KGC model to $f_{\mathcal{G}_k^+}^{APL}(\cdot)$. This manner addresses the domain shift issue [51] of joint method by incorporating the strengths of transfer learning.

We opt for a simple MLP as the framework of the KGC model and optimize it by minimizing the BCE loss $\mathcal{L}_{kgc}$:

$$\mathcal{L}_{kgc} = \frac{1}{|\widetilde{\mathcal{G}}_k^-|}\sum_{i=1}^{|\widetilde{\mathcal{G}}_k^-|} y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i), \tag{10}$$

$$\hat{y}_i = \sigma(f_{KGC}(\widetilde{\mathcal{G}}_k^{-,i};\theta_{KGC})) = \sigma(MLP([\mathbf{e}_h^i||\mathbf{e}_r^i||\mathbf{e}_t^i])), \tag{11}$$

$$\widetilde{\mathcal{G}}_k^- = [\mathcal{G}_k^- || \mathcal{G}_k^{-,neg}], \tag{12}$$

where σ is the sigmoid function, y_i is the ground truth of the i-th triplet of $\widetilde{\mathcal{G}}^-$, $||$ is the concatenate operation, and θ_{KGC} are the trainable parameters of KGC model $f_{KGC}(\cdot)$. $\mathbf{e}_h^i, \mathbf{e}_r^i, \mathbf{e}_t^i \in \mathbb{R}^d$ are the embeddings of head entity h, relation r, and tail entity t in the i-th triplet of $\widetilde{\mathcal{G}}^-$, respectively. $\mathcal{G}_k^{-,neg}$ consists of spurious triplets that are constructed by randomly replacing either head entity h, relation r, or tail entity t based on $\mathcal{G}_k^-$. Hence, the label of triplets in $\mathcal{G}_k^{-,neg}$ is 0. Conversely, triples in $\mathcal{G}_k^-$ are labeled as 1. After the $f_{KGC}(\cdot)$ has converged, we employ it to measure the authenticity of triplets in the potential KG $\mathcal{G}_k^+$:

$$\mathcal{Q}_k^+ = f_{KGC}(\mathcal{G}_k^+; \theta_{KGC}), \tag{13}$$

where $\mathcal{Q}_k^+ \in \mathbb{R}^{|\mathcal{G}_k^+|}$ is the attribute score vector of the potential KG $\mathcal{G}_k^+$, $|\mathcal{G}_k^+|$ is the number of triplets of $\mathcal{G}_k^+$.

$\mathcal{Q}_k^+ \in \mathbb{R}^{|\mathcal{G}_k^+|}$ in Eq (9) and $\mathcal{P}_k^+ \in \mathbb{R}^{|\mathcal{G}_k^+|}$ in Eq (13) are the attribute score vectors of the potential KG $\mathcal{G}_k^+$ for authenticity and task relevance, respectively. Therefore, freezing the well-trained $f_{KGC}(\cdot)$, we align $\mathcal{Q}_k^+$ and $\mathcal{P}_k^+$ by minimizing the Maximum Mean Discrepancy (MMD) loss [9], thereby transferring the expertise of $f_{KGC}(\cdot)$ to $f_{\mathcal{G}_k^+}^{APL}(\cdot)$. This process is formally defined as follow:

$$\mathcal{L}_{mmd} = \sup_{||g||_{\mathcal{H}} \leq 1} (\mathbb{E}[g(\sigma(\mathcal{Q}_k^+))] - \mathbb{E}[g(\sigma(\mathcal{P}_k^+))]), \tag{14}$$

where σ is sigmoid function. We take the gaussian kernel function $g(x) = \exp(-\frac{||x-\mu||}{2\sigma^2})$ as the mapping function in $\mathcal{L}_{mmd}$, and constrain its norm less than or equal to 1 in reproducing kernel Hilbert space $\mathcal{H}$ [1].

By jointly optimizing $f_{\mathcal{G}_k^+}^{APL}(\cdot)$ through minimizing the MMD loss $\mathcal{L}_{mmd}$ and recommendation loss $\mathcal{L}_{rec}$ (Eq (24)), $f_{\mathcal{G}_k^+}^{APL}(\cdot)$ is able to assess both the task relevance and authenticity of triplets in the potential KG $\mathcal{G}_k^+$. It is worth noting that the Expertise Transfer Mechanism allows EditKG to avoid reliance on the KGC model during the inference stage, making EditKG easier to deploy.

Thereafter, we incorporate $f_{\mathcal{G}_k^+}^{APL}(\cdot)$ with Gumbel-Max reparameterization trick to obtain the refined attribute score vector $\mathcal{M}_k^+$ for the potential KG $\mathcal{G}_k^+$:

$$\widetilde{m}_k^i = \mathbb{I}[\sigma(\widetilde{p}_k^i + \log(\epsilon) - \log(1-\epsilon)) > \mu], \tag{15}$$

$$\widetilde{p}_k^i = f_{\mathcal{G}_k^+}^{APL}(\mathcal{G}_k^{+,i}; \theta_{\mathcal{G}_k^+}), \tag{16}$$

where $\epsilon \in \text{Uniform}(0,1)$ is a random variable, and $\mathbb{I}[\cdot]$ is the indicator function, the same as in Eq (8). $\mathcal{M}_k^+ = [\widetilde{m}_k^1, \cdots, \widetilde{m}_k^{|\mathcal{G}_k^+|}]$ is the refined attribute score vector of potential KG $\mathcal{G}_k^+$. $\widetilde{m}_k^i > 0$ indicates that the i-th triplet of $\mathcal{G}_k^+$ is a supplementary attribute with authenticity and task relevance. Conversely, $\widetilde{m}_k^i = 0$ denotes that the i-th triplet of $\mathcal{G}_k^+$ is a supplementary attribute with spuriousness or task irrelevance. We integrate the potential KG $\mathcal{G}_k^+$ with its corresponding $\mathcal{M}_k^+$ to form a purified potential KG $\widetilde{\mathcal{G}}_k^+ = \{(h,r,t,m^{(h,r,t)}) | (h,r,t) \in \mathcal{G}_k^+, m^{(h,r,t)} \in \mathcal{M}_k^+\}$.

Finally, we integrate the purified primary KG $\widetilde{\mathcal{G}}_k$ with the purified potential KG $\widetilde{\mathcal{G}}_k^+$, to obtain the balanced KG $\mathcal{G}_k^b$:

$$\mathcal{G}_k^b = [\widetilde{\mathcal{G}}_k || \widetilde{\mathcal{G}}_k^+]. \tag{17}$$

Each tuple $(h,r,t,m^{h,r,t}) \in \mathcal{G}_k^b$ contains the triplet (h,r,t) and corresponding attribute score $m^{h,r,t}$. $m^{(h,r,t)} = 0$ denotes that the corresponding triplet (h,r,t) is a spurious or task-irrelevant attribute.

3.3 Aggregator and Prediction

Based on the balanced KG $\mathcal{G}_k^b$, we utilize the LightGCN [11] to learn the knowledge representation for items. Specifically, to highlight the supplementary item attributes within the purified potential KG $\widetilde{\mathcal{G}}_k^+$, similar to [56], we transfer $\mathcal{G}_k^b$ into two parts: $\mathcal{G}_k^{b,*} = \{\mathcal{T}_1, \mathcal{T}\}$, $\mathcal{G}_k^{b,+} = \{\mathcal{T}_2, \mathcal{T}\}$, where $\mathcal{T}_1 \cup \mathcal{T}_2 \cup \mathcal{T} = \mathcal{G}_k^b$. $\mathcal{T}_1$ and $\mathcal{T}_2$ correspond to the one-hop attributes of items in the purified primary KG $\widetilde{\mathcal{G}}_k$ and the purified potential KG $\widetilde{\mathcal{G}}_k^+$, i.e., $\mathcal{T}_1 \subset \widetilde{\mathcal{G}}_k$ and $\mathcal{T}_2 = \widetilde{\mathcal{G}}_k^+$. $\mathcal{T}$ correspond to the high-order (i.e., > 1) attributes of items in the purified primary KG $\widetilde{\mathcal{G}}_k$, i.e., $\mathcal{T} \subset \widetilde{\mathcal{G}}_k$. Next, based on $\mathcal{G}_k^{b,*}$ and $\mathcal{G}_k^{b,+}$, we conduct the representation aggregation procedure, which is defined as follows:

$$\mathbf{e}_h = \sum_{l=0}^{L} [\mathbf{e}_h^{+,(l)} || \mathbf{e}_h^{*,(l)}], \tag{18}$$

$$\mathbf{e}_h^{*,(l)} = \frac{1}{|\mathcal{G}_k^{b,*}|} \sum_{(h,r,t,m^{(h,r,t)}) \in \mathcal{G}_k^{b,*}} (\mathbf{e}_t^{*,(l-1)} + \mathbf{e}_r^*) \cdot m^{(h,r,t)}, \tag{19}$$

$$\mathbf{e}_h^{+,(l)} = \frac{1}{|\mathcal{G}_k^{b,+}|} \sum_{(h,r,t,m^{(h,r,t)}) \in \mathcal{G}_k^{b,+}} (\mathbf{e}_t^{+,(l-1)} + \mathbf{e}_r^+) \cdot m^{(h,r,t)}, \tag{20}$$

where $||$ represents the concatenate operation. We apply the mean pooling on the results of L GCN layers to obtain the final knowledge representation of entity h, i.e., $\mathbf{e}_h \in \mathbb{R}^{2d}$. We use $\mathbf{e}_i$ to denote the knowledge representation of item v_i. Then, we concatenate the $\mathbf{e}_i$ and the embedding $\mathbf{h}_i^v$ to obtain the comprehensive representation $\hat{\mathbf{h}}_i^v$ of item v_i, where $\hat{\mathbf{h}}_i^v \in \mathbb{R}^{3d}$.

Then, we utilize the comprehensive item representation and the user-item interaction graph $\mathcal{G}$ to obtain the final user and item embeddings, thereby making recommendation:

$$y_i^j = \frac{\exp(\widetilde{\mathbf{h}}_i^u \widetilde{\mathbf{h}}_j^{v^{\mathrm{T}}})}{\sum_{v_k \in \mathcal{V}} \exp(\widetilde{\mathbf{h}}_i^u \widetilde{\mathbf{h}}_k^{v^{\mathrm{T}}})}, \tag{21}$$

$$\widetilde{\mathbf{h}}_i^u = \mathbf{h}_i^u + \overline{\mathbf{h}}_i^u; \;\; \widetilde{\mathbf{h}}_i^v = \hat{\mathbf{h}}_i^v + \overline{\mathbf{h}}_i^v \tag{22}$$

$$\overline{\mathbf{h}}_i^u = \frac{1}{|\mathcal{N}_u^i|} \sum_{v_j \in \mathcal{N}_u^i} \hat{\mathbf{h}}_j^v; \;\; \overline{\mathbf{h}}_i^v = \frac{1}{|\mathcal{N}_v^i|} \sum_{u_j \in \mathcal{N}_v^i} \mathbf{h}_j^u, \tag{23}$$

where $\mathbf{h}_i^u \in \mathbb{R}^{3d}$ is the initial embedding of user u_i, $\widetilde{\mathbf{h}}_i^u, \widetilde{\mathbf{h}}_i^v \in \mathbb{R}^{3d}$ are the final embeddings of user u_i and item v_i, respectively. $\mathcal{N}_u^i$ denotes the items that user u_i has interacted with, and $\mathcal{N}_v^i$ denotes the users that item v_i has interacted with. y_i^j represents the probability of user u_i interacting with item v_j.

3.4 Model Optimization

We utilize cross-entropy as the loss function to optimize the parameters of EditKG:

$$\mathcal{L}_{rec} = \frac{1}{N} \sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1-y_i) \log(1-\hat{y}_i), \tag{24}$$

where $\hat{y}_i$ denotes the one-hot encoding vector of the ground truth, and N is the number of train data. We conduct the label smoothing strategy [20] on $\hat{y}_i$ to alleviate overfitting.

3.4.1 ***Gradient Match Strategy.*** As described in section 3.2.2, the update of the parameters $\theta_{\mathcal{G}_k^+}$ of the second APL $f_{\mathcal{G}_k^+}^{APL}(\cdot)$ is supervised by both $\mathcal{L}_{mmd}$ and $\mathcal{L}_{rec}$. The update process of $\theta_{\mathcal{G}_k^+}$ is described as:

$$\theta_{\mathcal{G}_k^+} = \theta_{\mathcal{G}_k^+} - \eta(\nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+}) + \nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+})), \tag{25}$$

where η is learning rate, $\nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+})$ and $\nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+})$ are the gradients of $\theta_{\mathcal{G}_k^+}$ with respect to the $\mathcal{L}_{mmd}$ and $\mathcal{L}_{rec}$, respectively.

However, we encounter the gradient conflict problem identified by Yu et al. [47]. Specifically, we observe that the gradients $\nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+})$ and $\nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+})$ point away from each other during the update process of $\theta_{\mathcal{G}_k^+}$. This results in an unstable learning process for $\theta_{\mathcal{G}_k^+}$, consequently leading to a suboptimal recommendation performance. To address the gradient conflict problem, we propose the Gradient Match Strategy.

Inspired by PCGrad [47], this strategy project gradient $\nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+})$ onto the normal vector of gradient $\nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+})$ if encountering the gradient conflict. Therefore, Eq (25) can be rewritten as:

$$\theta_{\mathcal{G}_k^+} = \begin{cases} \theta_{\mathcal{G}_k^+} - \eta(\nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+}) + \nabla\widetilde{\mathcal{L}}_{mmd}(\theta_{\mathcal{G}_k^+})) & if: \omega < 0, \\ \theta_{\mathcal{G}_k^+} - \eta(\nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+}) + \nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+})) & if: \omega \geq 0, \end{cases} \tag{26}$$

$$\nabla\widetilde{\mathcal{L}}_{mmd}(\theta_{\mathcal{G}_k^+}) = \nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+}) - \frac{\nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+}) \cdot \nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+})}{||\nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+})||^2}\nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+}), \tag{27}$$

$$\omega = \nabla\mathcal{L}_{rec}(\theta_{\mathcal{G}_k^+}) \cdot \nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+}), \tag{28}$$

where $\omega < 0$ denotes the update process of parameter $\theta_{\mathcal{G}_k^+}$ encountering the gradient conflict problem, and we adjust $\nabla\mathcal{L}_{mmd}(\theta_{\mathcal{G}_k^+})$ to $\nabla\widetilde{\mathcal{L}}_{mmd}(\theta_{\mathcal{G}_k^+})$ by Eq (27) to avoid this issue. This enables $\theta_{\mathcal{G}_k^+}$ to update stably, leading to a powerful recommendation performance. Algorithm 1 demonstrates the overall algorithm of EditKG.

Table 1: Statistics of three datasets.

Stats.	Last-FM	Amazon-Book	Yelp2018
#Item	23,566	70,679	45,919
#User	48,123	24,915	45,538
#Interactions	3,034,796	846,434	1,183,610
#Density	2.7e-3	4.8e-4	5.7e-4
	Knowledge Graph		
#Entities	58,266	29,714	47,472
#Relations	9	39	42
#Triplets	464,567	686,516	869,603

Algorithm 1 Implementation of EditKG

Input: User-Item interaction Graph $\mathcal{G}$, Knowledge Graph $\mathcal{G}_k$, recommendation model $\mathcal{F}(\cdot)$, KGC model $f_{KGC}(\cdot)$.
Output: recommendation model $\mathcal{F}(\cdot)$ with optimal parameters Θ
randomly initialize parameters Θ and θ_{kgc} of $\mathcal{F}(\cdot)$ and $f_{KGC}(\cdot)$, respectively;
for every epoch **do**
  if epoch < 1 **then**
    Utilize $\mathcal{G}_k$ to calculate $\mathcal{L}_{kgc}$ and optimize θ_{kgc};
    Apply *Knowledge Generator* to obtain $\mathcal{G}_k^+$ rely on $p(\cdot)$ in Eq (1);
  else
    Utilize $\mathcal{G}_k^-$ to calculate $\mathcal{L}_{kgc}$ and optimize θ_{kgc};
    Apply *Knowledge Generator* to obtain $\mathcal{G}_k^+$ rely on $s(\cdot)$ in Eq (2);
  end if
  Dropping negative attributes in $\mathcal{G}_k^+$ and $\mathcal{G}_k$ via Knowledge Deleter;
  Learn representation of users and items through Aggregator;
  Make prediction and calculate $\mathcal{L}_{mmd}$ and $\mathcal{L}_{rec}$;
  Update Θ based on $\mathcal{L}_{mmd}$ and $\mathcal{L}_{rec}$;
  Extract $\mathcal{G}_k^-$ from Knowledge Deleter for KGC model $f_{KGC}(\cdot)$;
  Early stopping strategy;
end

4 EXPERIMENTS

We conduct comprehensive experiments to evaluate the performance of EditKG by answering following research questions:

- **RQ1:** How does EditKG perform compared with different types of recommendation methods?
- **RQ2:** How do different key designs of EditKG impact its overall performance?
- **RQ3:** What is the result of knowledge balancing of EditKG?
- **RQ4:** What is the efficacy of EditKG in mitigating the cold-start problem?
- **RQ5:** How does the Gradient Match Strategy impact EditKG? Besides, can EditKG provide an intuitive impression of explainability?

4.1 Experimental Settings

4.1.1 ***Datasets.*** To conduct diversified scenario testing, we select three distinct real-life datasets as benchmarks: *Last-FM* for music recommendation, *Amazon-Book* for product recommendation, and *Yelp2018* for business venue recommendation. Following the settings in previous work [36, 45, 56], we map items into Freebase [55] entities and collect the two-hop neighbors to build primary KG. We split data same as [34, 56] to construct training (i.e., 80%), validation (i.e., 10%), and test set (i.e., 10%). Table 1 provides a summary of the statistics for the three datasets, along with their primary KG.

4.1.2 ***Evaluation Metrics.*** Following in [36, 45, 56], we adopt two widely used evaluation metrics, i.e, Recall@N and NDCG@N, to evaluate the performance of EditKG and all baseline methods. We report the average results of evaluations for all users in the test set with the default setting $N = 20$.

4.1.3 ***Baselines.*** We compare the performance of EditKG with baseline methods following various research lines.

General Collaborative Filtering Methods.

- **BPR** [24] incorporates pairwise ranking loss with matrix factorization to make recommendation.
- **GC-MC** [3] proposes a graph auto-encoder based on differentiable message passing on the bipartite user-item graph.
- **LightGCN** [11] simplifies GCN by extracting its most essential component, neighborhood aggregation.
- **SGL** [39] applies dropout to generate multiple views of user-item graph and leverages self-supervised learning on them to enhance recommendation.

Embedding-based Knowledge-aware Recommenders.

- **CKE** [48] applies TransR [15] to capture the heterogeneous information and leverages auto-encoders to further enrich the representation of items from the knowledge base.
- **KTUP** [6] utilizes TransH [38] to mine user's preference at a finer granularity. Next, it jointly trains it with a KG completion model combining several transfer schemes.

GNN-based Knowledge Graph-aware Recommenders.

- **KGCN** [33] learns the entity representation by incorporating neighborhood information bias into aggregating message.
- **KGAT** [34] recursively propagates the weighted neighborhood information by combining the attention mechanism to refine the node's embedding.
- **KGIN** [36] models the intent behind a user-item interaction as an attentive combination of KG relations and recursively integrates the relation sequences for better representation.
- **KGCL** [45] applies graph augmentation technique and contrastive learning to suppress the noise of KG and amplify the unbiased user-item interactions.
- **KGRec** [44] integrates generative and contrastive self-supervised tasks to highlight useful knowledge and align signals from knowledge and user-item interaction views.
- **KRDN** [56] improves the performance and robustness of model by dropping the noise item attributes and implicit feedback.

Parameter Settings. Our proposed EditKG is implemented with PyTorch. For the hyperparameter settings, we fix ID embedding size d, batch size and learning rate to 128, 4096 and $5e^{-4}$, respectively. Based on the experiment results on the validation set, we set GNN layer L, η in Eq (4) and μ in Eq (7) to 2, 1.4, and 0.2, respectively. The model parameters are initialized by the Xavier initializer and optimized via Adam. For a fair comparison, we report the best performance of all baseline methods according to their published papers. For the unreported data, we evaluate their performance by provided code and default settings.

4.2 Performance Comparison (RQ1)

Table 2 reports the performance of all models, where *%Imp* denotes the percentage improvement of EditKG over the best performance baseline methods. Based on these results, we have the following observations.

Compared to various types of baseline methods, EditKG exhibits a significant superiority across all datasets. In particular, EditKG reaches the optimal performance in NDCG@20, showing percentage improvements of 15.75%, 9.62%, and 3.68% for Last-FM, Amazon-book, and Yelp2018, respectively. Similarly, EditKG achieves the best performance in Recall@20, and its percentage improvements are similar to NDCG@20. These enhancements account for two reasons: Firstly, the Knowledge Generator generates comprehensive potential attributes, providing strong support to ensure the sufficient representation of items. Secondly, the Knowledge Deleter effectively drops the spurious and task-irrelevant attributes in the potential KG and the primary KG, resulting in a balanced KG for better recommendation.

A further analysis of EditKG across three datasets demonstrates that the greatest improvement occurs in Last-FM, while Yelp2018 shows the least. This is because, compared to the KG associated with Yelp2018, the KGs for LastFM and Amazon-Book are more sparse, leading to a more severe knowledge imbalance. This issue is effectively alleviated by EditKG.

By analyzing the performance of various types of baseline methods, we observe that KG-aware methods typically outperform general collaborative filtering methods. This is because KGs alleviate the cold-start problem by providing external information for items. Additionally, among these KG-aware methods, GNN-based methods generally outperform embedding-based methods. It is because KGs and user-item interactions exhibit complex topological characteristics, while GNNs are able to modeling such complex topological characteristics. Furthermore, KRDN, KGRec, and KGCL generally outperform other baselines, as they apply contrastive learning to provide additional supervision signals in learning item embeddings.

Table 2: Overall performance comparison. In each column, the highest performing result is marked in bold, and the second highest is indicated with an underline. The superscript † denotes that the improvement is statistically significant, with p-value < 0.01.

MODEL	Last-FM		Amazon-Book		Yelp2018	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
BPR	0.0690	0.0585	0.1244	0.0658	0.0568	0.0462
GC-MC	0.0709	0.0631	0.1283	0.0670	0.0688	0.0453
LightGCN	0.0738	0.0647	0.1398	0.0736	0.0632	0.0519
SGL	0.0879	0.0775	0.1445	0.0766	0.0788	0.0518
CKE	0.0845	0.0718	0.1375	0.0685	0.0653	0.0423
KTUP	0.0865	0.0671	0.1401	0.0692	0.0672	0.0411
KGCN	0.0879	0.0694	0.1111	0.0569	0.0532	0.0338
KGAT	0.0870	0.0743	0.1390	0.0739	0.0705	0.0463
KGIN	0.0978	0.0848	0.1687	0.0915	0.0698	0.0451
KGCL	0.0905	0.0769	0.1496	0.0793	0.0748	0.0491
KGRec	0.0943	0.0810	0.1443	0.0782	0.0745	0.0478
KRDN	0.1023	0.0946	0.1574	0.0886	0.0842	0.0544
EditKG	**0.1212**†	**0.1095**†	**0.1770**†	**0.1003**†	**0.0854**†	**0.0564**†
%Imp	18.48%	15.75%	4.92%	9.62%	1.43%	3.68%

Table 3: Ablation Study

Variant Models	Last-FM		Amazon-Book		Yelp2018	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
w/o KGN	0.1022	0.0943	0.1619	0.0874	0.0838	0.0540
w/o KDL	0.1101	0.1025	0.1660	0.0912	0.0844	0.0547
w/o KGC	0.1153	0.1065	0.1713	0.0964	0.0846	0.0548
w/o APL	0.1150	0.1061	0.1714	0.0963	0.0845	0.0546
w/o KTM	0.1189	0.1078	0.1749	0.0981	0.0847	0.0558
w/o MIC	0.1114	0.1037	0.1688	0.0916	0.0842	0.0545
w/o SMC	0.1161	0.1066	0.1727	0.0968	0.0850	0.0556
EditKG	**0.1212**	**0.1095**	**0.1770**	**0.1003**	**0.0854**	**0.0564**

4.3 Ablation Study (RQ2)

In this subsection, we first conduct ablation experiments on the Knowledge Generator and the Knowledge Deleter of EditKG from a macro-level perspective. Specifically, we design the following variant models to compare with EditKG:

- **w/o KGN:** We remove the Knowledge Generator from EditKG and employ the Knowledge Deleter to drop task-irrelevant attributes in the primary KG.

- **w/o KDL:** We remove the Knowledge Deleter from EditKG and directly concatenate the primary KG and the potential KG to obtain a balanced KG.

Table 3 reports the experimental results. We can observe a significant decline in EditKG's performance upon removing the Knowledge Generator or the Knowledge Deleter (i.e., w/o KGN and w/o KDL). This highlights the effectiveness of the two components.

Besides, we further explore the effectiveness of the Knowledge Generator and the Knowledge Deleter from a micro-level perspective. The corresponding variant models are defined as follows:

- **w/o KGC:** We remove the KGC model and solely rely on the APL to drop the task-irrelevant attributes in the primary KG and the potential KG.
- **w/o APL:** We remove the APL and only rely on the KGC model to drop the spurious attributes in the potential KG.
- **w/o KTM:** We remove the Knowledge Transfer Mechanism and combine the KGC model with the APL to orderly drop the spurious and task-irrelevant attributes in the potential KG.
- **w/o MIC:** We remove the mutual information correlation from the Knowledge Generator and solely employ the semantic correlation to generate potential attributes.
- **w/o SMC:** We remove the semantic correlation from the Knowledge Generator and solely employ the mutual information correlation to generate potential attributes.

The experimental results are outlined in Table 3. Based on these results, we can draw the following conclusions: (i) Removing either the KGC model (i.e., w/o KGC) or the APL (i.e., w/o APL) from EditKG results in a performance decline. This is because the KGC model and the APL enable EditKG to drop spurious and task-irrelevant attributes, respectively. (ii) The performance of the variant model w/o KTM is worse than that of EditKG. This is due to the KTM effectively transfers the expertise of the KGC to the APL, thereby enabling EditKG to drop negative attributes more accurately. Besides, this mechanism allows EditKG to avoid reliance on the KGC model during the inference stage, making EditKG easier to deploy. (iii) EditKG outperforms the variant model w/o MIC and w/o SMC. The reason is that the mutual information correlation and semantic correlation depict the attribute correlation between items from static and dynamic perspectives, respectively. Their integration yields more precise potential attributes.

4.4 The Effect of Knowledge Balancing (RQ3)

To investigate the knowledge balancing result, we first extract the balanced KG from a well-trained EditKG. Then, we count the number of attributes for items in the primary KG and balanced KG. Meanwhile, we utilize the Savitzky-Golay filter [25] to smooth them, aim at highlighting their characteristic. Figure 3 shows the popularity of items and their corresponding number of attributes in the primary KG and the balanced KG in terms of Last-FM and Yelp2018. The shaded area denotes the gap in the number of attributes between the two KGs. Compared to the primary KG, the long-tail items acquire a significantly higher number of attributes from the balanced KG, indicating that EditKG effectively supplements the lacking attributes for these items. The number of attributes of the popular items has not been significantly increased; in fact, it has even declined. This result displays the effectiveness of EditKG in dropping the task-irrelevant and spurious attributes.

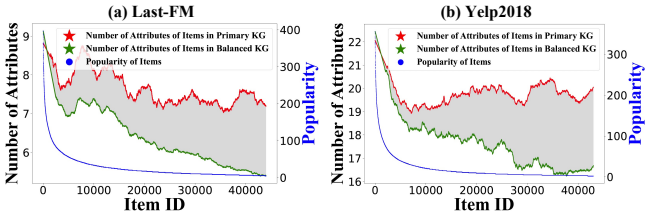

Figure 3: The popularity of items and their corresponding number of attributes in the primary KG and the balanced KG, as shown for (a) Last-FM and (b) Yelp2018.

4.5 Model Benefits Investigation in Alleviating Cold-Start (RQ4)

To explore the performance of EditKG in handling the cold-start problem, we conduct grouped experiments on users and items under various cold-start levels. In these experiments, the performance of EditKG is evaluated in comparison with three well-performing baseline models, i.e., KRDN, KGCL, and KGIN.

For the user cold-start test, we evenly divide users into five groups based on the number of interactions, with larger group numbers indicating a stronger user cold-start state. Figure 4 (a) presents the experiment results on Last-FM in terms of Recall@20. The results demonstrate the superiority of EditKG across all user groups, with its advantages becoming more pronounced in groups with higher degrees of cold-start. For the item cold-start test, we calculate the average sparsity for each user based on the popularity of items with which they have interacted. Then, we divide the users into five groups based on this average sparsity. A larger group number indicates a stronger item cold-start state. Figure 4 (b) displays the experiment results on Last-FM in terms of Recall@20. The results indicate that EditKG exhibits greater superiority compared to the baseline models, especially in item groups with high cold-start degree. Overall, these experiment results demonstrate the efficacy of EditKG in alleviating the cold-start issue.

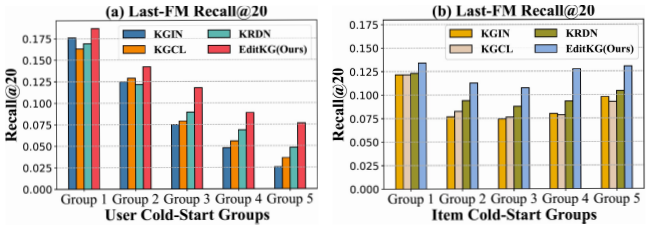

Figure 4: The performance of EditKG and three baselines across different user cold-start groups (a) and item cold-start groups (b) in Last-FM.

4.6 Visualization of Model Convergence Process and Case Study (RQ5)

To assess the impact of Gradient Match Strategy (GM) on EditKG, we record the changes in total loss ($\mathcal{L}_{rec} + \mathcal{L}_{mmd}$) of two models, named EditKG and EditKG without GM, during their training

process on the Amazon-Book. Concurrently, we record the corresponding changes of in their performance. These data are shown in Figure 5 (a). We observe that the Gradient Match Strategy not only accelerates the convergence of EditKG, leading to a lower total loss value, but also enhances the performance of EditKG. This is because Gradient Matching Strategy effectively helps EditKG avoid the gradient conflict problem during training.

Moreover, to explore the interpretability of EditKG, we randomly select two items with attribute correlation score higher than η from the Amazon-Book. Then, we obtain their knowledge balancing results via EditKG and map these results to real names through Freebase. Figure 5 (b) demonstrates these results. Specifically, EditKG supplements the true-to-reality attributes, such as "original language" and "genre", and removes the supplementary spurious attributes "book author" and the task-irrelevant attributes "valuenotation is reviewed" for the unpopular book *Never Count Out The Dead*. Similarly, for the popular book *The Bottoms*, EditKG drops the existing task-irrelevant attribute "valuenotation is reviewed", and the supplementary spurious attribute "book author". The number of attributes for both books, originally 2 and 7 respectively, is adjusted to 6, resulting in a more balanced attribute distribution.

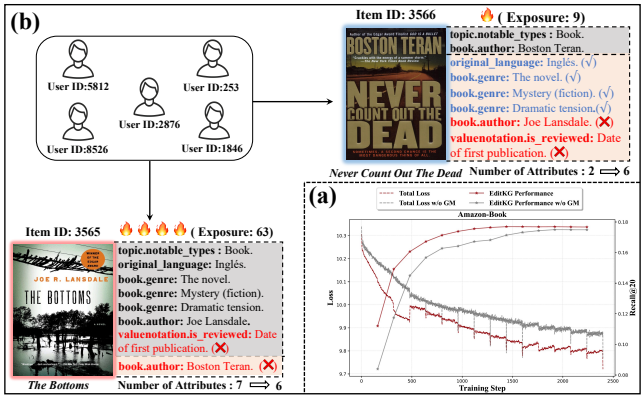

Figure 5: Training process visualization (a). A random example of knowledge balancing result of EditKG in Amazon-Book (b). The gray and orange backgrounds denote existing and supplementary attributes, respectively. Blue and red characters signify retained and dropped attributes, respectively.

5 RELATED WORK

5.1 Knowledge-aware Recommendation

Current KG-aware recommendation efforts can be generally categorized into four groups: (i) Embedding-based methods[6, 28, 32, 48, 49], they leverage transition-based techniques on KGs, e.g., TransE [4], to generate pretrained entity embeddings and link to items, thereby improving recommendation performance. For instance, CKE [48] leverages TransR to encode items' knowledge representation. KTUP [6] jointly trains recommendation model with TransH [38] to explore users' preference. (ii) Path-based methods[12, 31, 37, 40], which extract prior knowledge paths to refine the knowledge representation of items. For example, Ripplenet [31] incorporates the meta-paths based KG to iteratively extend users' potential interests. However, the definition of these prior knowledge paths often heavily relies on domain expertise and labor-intensive human efforts. (iii) Propagation-based methods [33, 34, 36, 56], they leverage GNNs to explore the high-order representation of items within KGs. For instance, KGAT [34] incorporates attention mechanism to learn the high-order knowledge representation of items. (iv) Contrastive learning-based methods [44, 45, 56, 57], which incorporate the strengths of self-supervised learning to enhance recommendation. For instance, KGCL [45] applies a contrastive learning paradigm to provide additional supervision signals for items through constructing items' multi-views, thereby enhancing the knowledge representation of items.

5.2 Cross-domain Recommendation

In KG-aware recommender systems, we consider KG as the source domain and the recommendation as the target domain. Under this perspective, KG-aware recommendation aligns with the paradigm of cross-domain recommendation, i.e., transfer valuable knowledge of the source domain to enhance recommendation in the target domain. Common techniques in cross-domain recommendation include transfer learning [21] and multi-task learning [53]. EditKG is inspired by works leveraging these techniques. For instance, Qiu et al. [23] develop a domain-aware attention network, which learns the importance of features cross different domains via attention mechanism. AFT [10] proposes a generative adversarial network to capture the transition pattern of information between different domains. Yang et al. [43] incorporate the strengths of knowledge distillation to extend supervised signals, thereby enhancing recommendation. Zhang et al. [52] propose a graphical modeling method to integrate the evidence with multiple forms, thereby enabling more accurate prediction of user preferences. STAR [27] mixes the recommendation data of cross-domains, so as to learn a comprehensive model to serve multiple domains.

6 CONCLUSION

In this paper, we propose the EditKG to alleviate the knowledge imbalance problem of KG-aware recommendation. To our knowledge, this is the first work to study this pervasive yet practical problem. The central concept of EditKG is to generate a balanced KG through editing KG, thereby alleviating the knowledge imbalance problem. The edit manners of EditKG include two parts: (i) Generating potential attributes through exploring the attribute correlations between items, and (ii) dropping spurious and task-irrelevant attributes via aligning attribute score and minimizing recommendation loss. Experimental results demonstrate that EditKG significantly outperforms state-of-the-art methods. In future work, we tend to explore the high-order lacking attributes of items in KGs, addressing the knowledge imbalance issue in a general way. Additionally, given that EditKG is only suited for static user-item interactions, we plan to refine EditKG in the future to accommodate the dynamic changes in online user-item interactions.

ACKNOWLEDGMENTS

This work was partially supported by National Key R&D Program of China (No.2022YFB3904204), NSF China (No.62272301, 61960206002, 42050105, 62020106005, 62061146002, 623B2071) and Shanghai Pilot Program for Basic Research - Shanghai Jiao Tong University.

REFERENCES

[1] Nachman Aronszajn. 1950. Theory of reproducing kernels. *Transactions of the American mathematical society* 68, 3 (1950), 337–404.

[2] Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Chi Kit Cheung. 2022. Why Exposure Bias Matters: An Imitation Learning Perspective of Error Accumulation in Language Generation. *arXiv preprint arXiv:2204.01171* (2022).

[3] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).

[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems* 26 (2013).

[5] Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL* 30 (2009), 31–40.

[6] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *The world wide web conference*. 151–161.

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[8] Yongjun Chen, Zhiwei Liu, Jia Li, Julian McAuley, and Caiming Xiong. 2022. Intent contrastive learning for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*. 2172–2182.

[9] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *The Journal of Machine Learning Research* 13, 1 (2012), 723–773.

[10] Xiaobo Hao, Yudan Liu, Ruobing Xie, Kaikai Ge, Linyao Tang, Xu Zhang, and Leyu Lin. 2021. Adversarial feature translation for multi-domain recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2964–2973.

[11] 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*. 639–648.

[12] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1531–1540.

[13] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems* 33, 2 (2021), 494–514.

[14] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. Melu: Meta-learned user preference estimator for cold-start recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1073–1082.

[15] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 29.

[16] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. 2018. Conditional adversarial domain adaptation. *Advances in neural information processing systems* 31 (2018).

[17] Yuanfu Lu, Yuan Fang, and Chuan Shi. 2020. Meta-learning on heterogeneous information networks for cold-start recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1563–1573.

[18] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. 2020. Parameterized explainer for graph neural network. *Advances in neural information processing systems* 33 (2020), 19620–19631.

[19] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2016. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712* (2016).

[20] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *Advances in neural information processing systems* 32 (2019).

[21] Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2009), 1345–1359.

[22] Ciyuan Peng, Feng Xia, Mehdi Naseriparsa, and Francesco Osborne. 2023. Knowledge graphs: Opportunities and challenges. *Artificial Intelligence Review* (2023), 1–32.

[23] Minghui Qiu, Bo Wang, Cen Chen, Xiaoyi Zeng, and Jun Huang. [n. d.]. Transfer Learning with Domain-aware Attention Network for Item Recommendation in E-commerce. ([n. d.]).

[24] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[25] Abraham Savitzky and Marcel JE Golay. 1964. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry* 36, 8 (1964), 1627–1639.

[26] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks* 20, 1 (2008), 61–80.

[27] Xiang-Rong Sheng, Liqin Zhao, Guorui Zhou, Xinyao Ding, Binding Dai, Qiang Luo, Siran Yang, Jingshan Lv, Chi Zhang, Hongbo Deng, et al. 2021. One model to serve all: Star topology adaptive recommender for multi-domain ctr prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4104–4113.

[28] Yu Tian, Yuhao Yang, Xudong Ren, Pengfei Wang, Fangzhao Wu, Qian Wang, and Chenliang Li. 2021. Joint knowledge pruning and recurrent graph convolution for news recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 51–60.

[29] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*. PMLR, 2071–2080.

[30] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*. PMLR, 2071–2080.

[31] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 417–426.

[32] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 world wide web conference*. 1835–1844.

[33] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge graph convolutional networks for recommender systems. In *The world wide web conference*. 3307–3313.

[34] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 950–958.

[35] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.

[36] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. 2021. Learning intents behind interactions with knowledge graph for recommendation. In *Proceedings of the web conference 2021*. 878–887.

[37] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5329–5336.

[38] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 28.

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

[40] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Xiyue Zhang, Hongsheng Yang, Jian Pei, and Liefeng Bo. 2021. Knowledge-enhanced hierarchical graph transformer network for multi-behavior recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4486–4493.

[41] Zihao Xu, Hao He, Guang-He Lee, Yuyang Wang, and Hao Wang. 2022. Graph-relational domain adaptation. *arXiv preprint arXiv:2202.03628* (2022).

[42] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).

[43] Chenxiao Yang, Junwei Pan, Xiaofeng Gao, Tingyu Jiang, Dapeng Liu, and Guihai Chen. 2022. Cross-task knowledge distillation in multi-task recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 4318–4326.

[44] Yuhao Yang, Chao Huang, Lianghao Xia, and Chunzhen Huang. 2023. Knowledge Graph Self-Supervised Rationalization for Recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3046–3056.

[45] Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. 2022. Knowledge graph contrastive learning for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1434–1443.

[46] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems* 33 (2020), 5824–5836.

[47] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems* 33 (2020), 5824–5836.

[48] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In

Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 353–362.

[49] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 353–362.

[50] Lei Zhang, Shanshan Wang, Guang-Bin Huang, Wangmeng Zuo, Jian Yang, and David Zhang. 2019. Manifold criterion guided transfer learning via intermediate domain generation. *IEEE transactions on neural networks and learning systems* 30, 12 (2019), 3759–3773.

[51] Marvin Zhang, Henrik Marklund, Nikita Dhawan, Abhishek Gupta, Sergey Levine, and Chelsea Finn. 2021. Adaptive risk minimization: Learning to adapt to domain shift. *Advances in Neural Information Processing Systems* 34 (2021), 23664–23678.

[52] Yi Zhang and Jamie Callan. 2005. Combining multiple forms of evidence while filtering. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. 587–595.

[53] Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering* 34, 12 (2021), 5586–5609.

[54] Yongfeng Zhang, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Understanding the sparsity: Augmented matrix factorization with sampled constraints on unobservables. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. 1189–1198.

[55] Wayne Xin Zhao, Gaole He, Kunlin Yang, Hongjian Dou, Jin Huang, Siqi Ouyang, and Ji-Rong Wen. 2019. Kb4rec: A data set for linking knowledge bases with recommender systems. *Data Intelligence* 1, 2 (2019), 121–136.

[56] Xinjun Zhu, Yuntao Du, Yuren Mao, Lu Chen, Yujia Hu, and Yunjun Gao. 2023. Knowledge-refined Denoising Network for Robust Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. 362–371.

[57] Ding Zou, Wei Wei, Xian-Ling Mao, Ziyang Wang, Minghui Qiu, Feida Zhu, and Xin Cao. 2022. Multi-level cross-view contrastive learning for knowledge-aware recommender system. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1358–1368.