InBox: Recommendation with Knowledge Graph using Interest Box Embedding

Zezhong Xu
Zhejiang University
China Mobile(Zhejiang) Research Innovation Institute
xuzezhong@zju.edu.cn

Yincen Qu
Alibaba Group
Hangzhou, China
yincen.qyc@alibaba-inc.com

Wen Zhang
Zhejiang University
ZJU-Ant Group Joint Lab of Knowledge Graph
zhang.wen@zju.edu.cn

Lei Liang
Ant Group
Hangzhou, China
leywar.liang@antgroup.com

Huajun Chen
Zhejiang University
ZJU-Ant Group Joint Lab of Knowledge Graph
huajunsir@zju.edu.cn

ABSTRACT

Knowledge graphs (KGs) have become vitally important in modern recommender systems, effectively improving performance and interpretability. Fundamentally, recommender systems aim to identify user interests based on historical interactions and recommend suitable items. However, existing works overlook two key challenges: (1) an interest corresponds to a potentially large set of related items, and (2) the lack of explicit, fine-grained exploitation of KG information and interest connectivity. This leads to an inability to reflect distinctions between entities and interests when modeling them in a single way. Additionally, the granularity of concepts in the knowledge graphs used for recommendations tends to be coarse, failing to match the fine-grained nature of user interests. This homogenization limits the precise exploitation of knowledge graph data and interest connectivity. To address these limitations, we introduce a novel embedding-based model called InBox. Specifically, various knowledge graph entities and relations are embedded as points or boxes, while user interests are modeled as boxes encompassing interaction history. Representing interests as boxes enables containing collections of item points related to that interest. We further propose that an interest comprises diverse basic concepts, and box intersection naturally supports concept combination. Across three training steps, InBox significantly outperforms state-of-the-art methods like HAKG and KGIN on recommendation tasks. Further analysis provides meaningful insights into the variable value of different KG data for recommendations.

PVLDB Reference Format:
Zezhong Xu, Yincen Qu, Wen Zhang, Lei Liang, and Huajun Chen. InBox: Recommendation with Knowledge Graph using Interest Box Embedding. PVLDB, 17(13): 4641 - 4654, 2024.
doi:10.14778/3704965.3704972

Wen Zhang and Huajun Chen are the corresponding authors.

Proceedings of the VLDB Endowment, Vol. 17, No. 13 ISSN 2150-8097.
doi:10.14778/3704965.3704972

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/zjukg/InBox.

1 INTRODUCTION

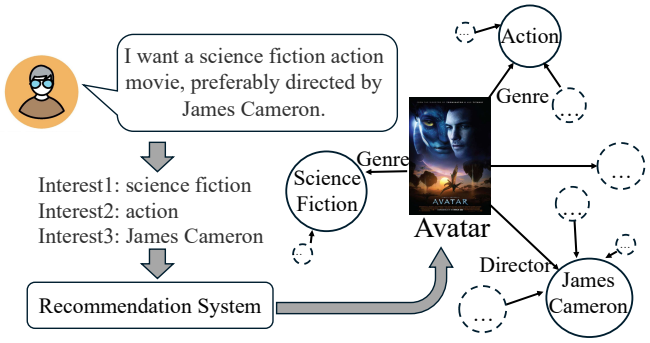

Figure 1: An example illustrating a specific interest is the combination of several concepts.

As the Internet develops, recommendation systems have emerged as vital tools for identifying user preferences across diverse domains, encompassing search engines, e-commerce platforms, and media services. Conventional recommendation approaches [10, 12, 14] primarily focus on collaborative filtering (CF). Although proficient at discerning collaborative patterns, these techniques grapple with issues of data sparsity and cold-start dilemmas. In recent times, there has been a growing interest in incorporating knowledge graphs, replete with abundant entity and relational data, into recommender systems. Such integration not only enhances the precision of recommendation results but also addresses cold-start predicaments and bolsters the explainability of the recommendation pipeline.

Employing auxiliary information such as KGs revolves around procuring suitable representations for items and users. Certain techniques investigate the utilization of knowledge graph pathways to bolster user-item interactions, thereby offering interpretability for recommendation outcomes. Nevertheless, these approaches necessitate labor-intensive feature engineering, such as manually crafted meta-paths, and grapple with subpar transferability and fluctuating performance. Graph neural networks (GNN) have also gained traction in recommendation tasks. By implementing an information aggregation scheme, multi-hop neighbor data is amalgamated into node representations. Although effective, GNN-based models typically hinge on the quality of the knowledge graph; in real-world situations, KGs tend to be sparse and replete with noise, which curtails their efficacy. Some of the early studies [2, 23, 34] incorporated item embeddings, derived from KG embedding (KGE) methodologies (e.g., TransE [4] and TransR [16]), as prior embeddings to augment the recommendation endeavor. While embedding-centric models profit from the straightforwardness and expressivity of KGE, they falter in capturing the higher-order dependencies between knowledge and user interests.

Indeed, the crux of recommendation lies in discerning user preferences and suggesting suitable items accordingly (e.g., as manifested by user embeddings in certain studies). While previously discussed approaches yield commendable results within knowledge-aware contexts, we contend that they fall short in adequately representing user interests when taking the subsequent two factors into account.

Firstly, user interests often encompass a wide range of related items, a complexity not fully addressed by traditional recommender systems. These systems typically represent user interests or profiles as singular points in the embedding space, which can be limiting when trying to establish connections between a user's preferences and a diverse set of items. For example, consider a user interested in movies like Figure1, and their interest is defined by a single attribute such as *directed by James Cameron*. This attribute alone can relate to multiple movies, illustrating the one-to-many relationship between an attribute and the movies it describes. This type of relationship is common in recommendation KGs, where items and their descriptive attributes form distinct entity sets, a feature more pronounced than in general KGs. Existing methods may not have considered this distinction in their modeling.

Furthermore, user interests are often composed of a combination of multiple coarse-grained concepts. Taking the domain of movie recommendations in Figure1 as an example, a user's interest might not be confined to a single genre such as *Science Fiction* or *Action*, but rather defined by a combination of these concepts, such as *Science Fiction Action*. This interest could further refine to include specific directorial styles, like *James Cameron's Science Fiction Action*. By consolidating concepts within KGs to construct the deterministic relationship between KG and interest definition, more accurate and personalized interests can be explicitly determined for distinct users. Unfortunately, existing models fall short of establishing a connection between the intersection of KG concepts and user preferences in a fine-grained way.

To address the aforementioned challenges, we introduce a novel model, InBox, wherein user interests are conceptualized as a box containing suitable items. In our paper, KG entities are categorized into two distinct groups: items and non-items. Items represent the

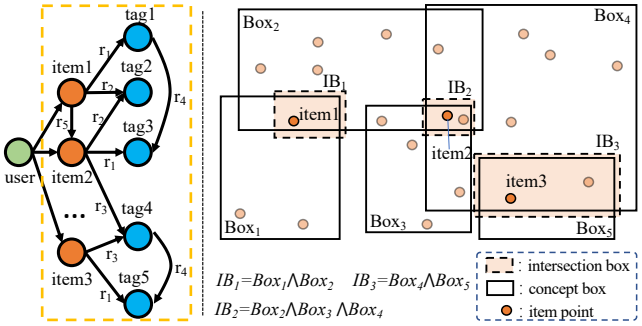

Figure 2: An illustration of the item points and concept boxes.

entities that are eligible for recommendation, poised to be presented to users as potential choices. On the other hand, non-items serve as descriptors of the items, providing attributes that enrich the understanding of the items but are not intended to be recommended to users. To illustrate this with an example from the realm of movie recommendations, the film *Avatar* qualifies as an item because it could be suggested to users as a viewing option. In contrast, the genre **Science Fiction**, which is an attribute of **Avatar**, acts as a descriptor or tag that helps categorize the movie but does not stand as a recommendation on its own. These two entity types are represented distinctly: An item is embedded into the vector space as a point, whereas a tag or relation is denoted by a box instead of a single point. For instance, in a movie recommendation context, consider the triplet *(Avatar, is directed by, James Francis Cameron)*: *Avatar* is represented as a point since it is an item, while the relation *is directed by* and the tag *James Francis Cameron* are both depicted as boxes; the relation-tag pair can be interpreted as a concept (since a tag linked with different relations could express different meanings. Like *(is directed by, James Francis Cameron)* and *(is written by, James Francis Cameron)*), although the tags in are both *James Francis Cameron*, these are two different concepts.

From a fundamental perspective, using box embeddings in combination with point embeddings instead of relying solely on point embeddings for modeling in recommendation systems offers several key advantages. Box embeddings can naturally capture hierarchical and overlapping relationships between entities and concepts, where larger boxes can subsume smaller ones, representing the subsuming of more specific concepts by more general ones. Moreover, the intersection or overlap between boxes can represent the shared characteristics or common aspects between different entities or concepts. Box embeddings are particularly well-suited for logical reasoning tasks, as they can encode logical operations such as intersection through geometric operations on the corresponding boxes, enabling more principled and interpretable reasoning within the embedding space. Additionally, by representing tags as regions rather than single points, box embeddings can inherently capture uncertainty and fuzziness in the data, where the size and shape of the boxes can encode the degree of uncertainty or fuzziness associated with a particular entity or relation, a desirable property in many real-world applications where data is noisy or imprecise.

In addition to defining the aforementioned representations, the model must possess the following capabilities: 1) ensuring that the

representations between items and tags satisfy their corresponding relationships within the KG; 2) defining the concept combination operation; 3) empowering the model to abstract user interests based on existing information. To achieve this, the model training process consists of three stages. The first stage focuses solely on the knowledge graph to obtain suitable representations that position item points within corresponding concept boxes. The second stage strives to situate items within the intersection box region of their associated concepts. The third stage employs interaction history to generate an interest box embedding, which is subsequently utilized to recommend potential items for users. To accomplish these stages, various distance functions are employed, encompassing point-to-point, box-to-box, and point-to-box distances.

To substantiate the efficacy of InBox, experiments are executed on four real-world datasets. Experimental outcomes demonstrate that InBox surpasses state-of-the-art models, including KGAT [26], KGIN [27], and HAKG [9], across all datasets. Additionally, model analysis reveals that different triplet types in KGs exert varying impacts on our model, potentially offering novel insights into the significance of KG triplets in KG-enhanced recommender systems.

In summary, our contributions to this work are as follows:

- We introduce a novel approach to modeling user interests for recommendation tasks, positing that interests can be represented by the conjunction of several general concepts in KGs.
- We present InBox, an innovative recommendation model that employs box embeddings to represent tags, relations, and user interests, fulfilling the criteria that an interest corresponds to a set of items and supports intersection logical operations.
- Through experimentation, we prove that InBox surpasses existing methods. Our model analysis unveils a fresh perspective for assessing the significance of various triplet types in knowledge-aware recommender systems.

2 PROBLEM FORMULATION

In this section, we introduce the data in the KG-enhanced recommender system, which comprises the user-item interaction history and various triplet types in the knowledge graph, as well as the task definition.

User-Item Interaction Graph. Given a user set $\mathcal{U}$ and an item set $\mathcal{I}$, the user-item interaction graph can be constructed as $\mathcal{G}_u = (u, i)|u \in \mathcal{U}, i \in \mathcal{I}$, where each (u, i) pair signifies that user u has previously engaged with item i, as determined by implicit feedback [20] from u to i. Note In this paper, we do not differentiate between various user behaviors (e.g., clicks, purchases); the datasets and baselines employed in our experiments also refrain from making such distinctions.

Auxiliary Knowledge Graph. KGs store structured information, including item attributes. Renowned KGs like Wordnet [18] and Freebase [3] have been developed. Let $\mathcal{E}$ represent a set of entities and $\mathcal{R}$ a set of relations. A knowledge graph can be defined as $\mathcal{G}_k = (e_1, r, e_2)|r \in \mathcal{R}, e_1, \in \mathcal{E}, e_2 \in \mathcal{E}$, where each triplet (e_1, r, e_2) indicates a relation r connects entities e_1 and e_2.

In the recommendation context, the entity set comprises the item set $\mathcal{I}$ and other non-item entities, which we term tags in this paper. Items are the entities earmarked for recommendation, ready to be offered to users as viable options. Conversely, non-items act as descriptors, furnishing attributes that enhance the comprehension of these items, yet they themselves are not candidates for recommendation. These non-item entities can be considered as a tag set $\mathcal{T}$. Accordingly, all the triplets in $\mathcal{G}_k$ can be categorized into three groups: (1) $(item, relation, item)$ triplet, denoted as *IRI* triplet; (2) $(tag, relation, tag)$ triplet, denoted as *TRT* triplet; (3) $(item, relation, tag)$ triplet, denoted as *IRT* triplet. Note that *(tag, relation, item)* is regarded as the same type as $(item, relation, tag)$ only if we use the inverse relation, so we do not specifically distinguish this type.

For instance, in the movie recommendation task, there are two items, *Avatar* and *Avatar*2, and two tags, *James Francis Cameron* and *America*. The examples of different triplet types are as follows:

- *(Avatar2, is sequel of, Avatar)* is an *IRI* triplet.
- *(James Francis Cameron, is citizen of, America)* is a *TRT* triplet.
- *(Avatar, is directed by, James Francis Cameron)* is an *IRT* triplet.

The relation-tag pair, such as *(is directed by, James Francis Cameron)*, can be viewed as a concept introduced earlier. The three distinct types of triplets will be processed differently in the training step.

Task Description. With the interaction graph $\mathcal{G}_u$ and KG $\mathcal{G}_k$, the task is learning a function predicting the relative possibility that a user would adopt an item he/she has not interacted with before.

3 METHODOLOGY

In this section, we introduce the details of the proposed model, and Figure 3 illustrates the architecture of the whole model. Firstly, we describe the representation format of items, tags, and relations in the KG. To train these different forms of embeddings and use them to get the recommendation results, we design three training steps to optimize the embeddings: (1) Basic pretraining, which aims to obtain suitable representations for each item, tag, and relation in the KG. To achieve this goal, three distinct distance functions are employed corresponding to the different triplet types; (2) Box intersection step, which seeks to embed each item point within the intersection region of its associated tag boxes. This meets our intuitive requirements when defining this representation; and (3) Interest box recommendation, which aspires to obtain the user interest box with the related tags' box embeddings, and employ the interest box embedding to get the predicted scores for all items the user has not interacted with, so that we can rank the items for recommendation results.

3.1 Item Point and Relation, Tag Box

To efficiently connect a set of items with a related concept in the embedding space, we first define point and box embeddings (i.e., axis-aligned hyper-rectangles) for the items and tags in the knowledge graph, respectively. Formally, in a vector space with dimension d, let each item be a point in the space, which could be represented as a vector $\mathbf{v} \in \mathbb{R}^d$. A tag in KG could be regarded as a box so that it could encompass multiple points in space that represent items. We define a box embedding as $\mathbf{b} = (\text{Cen}(\mathbf{b}), \text{Off}(\mathbf{b})) \in \mathbb{R}^{2d}$,

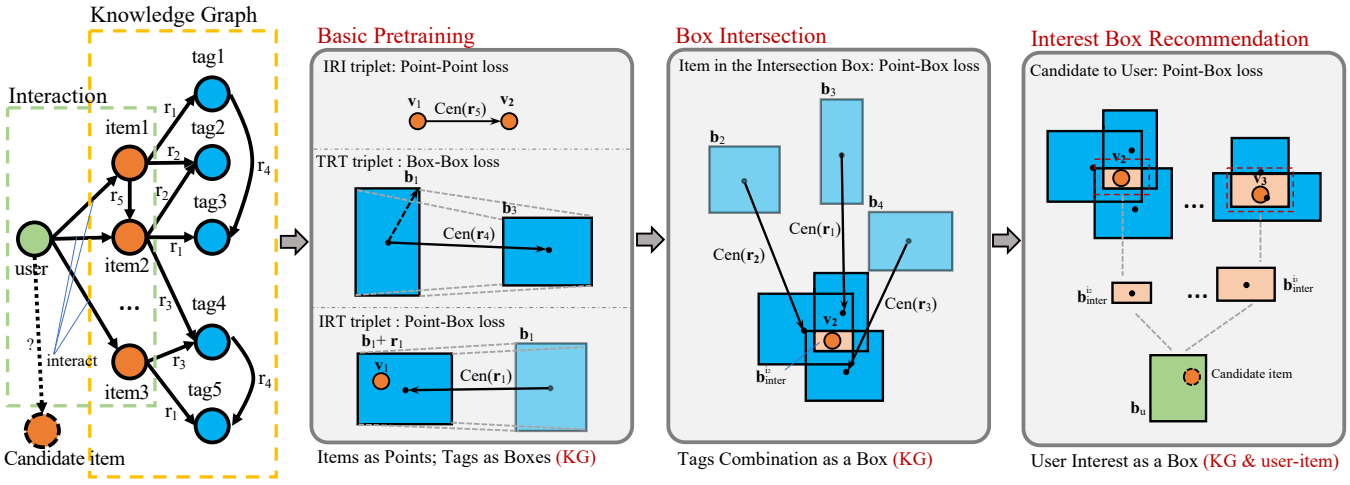

Figure 3: Framework of the proposed model InBox: The recommendation task is completed through three training steps (not all required). The initial two steps focus solely on the KG data to obtain suitable representations for items, tags, and relations. In the third step, the objective is to leverage the user's interest box to compute the matching score.

where $\text{Cen}(\mathbf{b})$ is the center point of the box, and $\text{Off}(\mathbf{b})$ is the offset embedding of the box, which indicates the size of the box. Through the calculation of two vectors, we can get the range of the box embedding on each dimension, so the box embedding $\mathbf{b}$ specifies the box range in the vector space as follows:

$$\text{Range}(\mathbf{b}) = \text{Cen}(\mathbf{b}) - \sigma(\text{Off}(\mathbf{b})) \preceq \mathbf{v} \preceq \text{Cen}(\mathbf{b}) + \sigma(\text{Off}(\mathbf{b})) \quad (1)$$

where σ is an activation function (e.g., ReLu) to make the $\text{Off}(\mathbf{b})$ be positive, and $\preceq$ is element-wise inequality. According to the definition, a box could contain all the items with their point embeddings inside the box range, building the connectivity of a concept and a set of items, which is one of the benefits of regarding a tag as a box embedding compared to a single point.

3.2 Basic Pretraining Step

The purpose of this step is to obtain a suitable initial representation for the entities and relations within the KG.

Building upon the definitions of box embedding and point embedding, we can map each item and tag in the KG as a point or a box in the space. As previously mentioned, the same tag with different relations may represent distinct concepts(like *(is directed by, James Francis Cameron)* and *(is written by, James Francis Cameron)*). To obtain appropriate concept embeddings, it is crucial to capture the effect of the relation on tags within the vector space. To achieve this, we have to define the representation of a relation as a box $\mathbf{b}_r = (\text{Cen}(\mathbf{b}_r), \text{Off}(\mathbf{b}_r)) \in \mathbb{R}^{2d}$, which is identical to a tag, and the relation box embedding will be used to modify the tag box embedding. When examining its association with the tag box, $\text{Cen}(\mathbf{b}_r)$ denotes the projection operator for the tag box's central point, while $\text{Off}(\mathbf{b}_r)$ signifies the resizing of the tag box's dimensions. Importantly, when evaluating the relation's impact on items, we exclusively employ $\text{Off}(\mathbf{b}_r)$, as a point lacks the notion of boundaries.

After the embedding is initialized randomly, it is necessary to get a representation that satisfies the triplets in KG. In the basic pretraining step, the main target is locating the item point inside each box of its related concept box, and the key idea is like the translation-based KG embedding models using relation as the projector. As we mentioned before, there are three kinds of triplets in the KG, which are *IRI* triplet, *TRT* triplet, and *IRT* triplet. Since each type of triplet contains a different head and tail entity embedding form, we design different ways of calculating the distance to get the final loss according to the type of the triplets, as Figure 4 illustrates the different calculation processes. For *IRI* triplets, we can simply use the point-to-point distance between the head and tail entities, as they are both represented as points. For *TRT* triplets, since both the head and tail entities are represented as boxes, we can use the box-to-box distance to calculate their similarity. Finally, for the *IRT* triplets, the head entity is a point, while the tail entity is a box, so in this case, we could use the point-to-box distance to measure their similarity. By calculating these distances, we can derive a loss function for each type of triplet. The goal of the basic pretraining step is to minimize the total loss, which will ensure that the item points are located inside the corresponding boxes and properly represent the facts in the KG.

IRI triplets. For this kind of triplet, the head entity and tail entity are both represented as points, and the object is to make the tail point close enough to the head point after projection with the relation. Let's take the example *(Avatar2, is the sequel of, Avatar)* in Section 2 as a demonstration. For this *IRI* triplet, the objective is to ensure that the object representation of *Avatar* closely aligns with the point representation of *Avatar2* after mapping by relation *is sequel of*. Specifically, for a triplet (h, r, t) (h and t are both items), the functional mapping with relation r is an element-wise addition from t to the target item h and gets the predicted embedding as:

$$\mathbf{v}'_h = \mathbf{v}_t + \text{Cen}(\mathbf{b}_r) \quad (2)$$

Note that the relation is utilized as a projector to the tail entity t for compliance with the other two kinds of triplets, which will be introduced later. Essentially, there is no difference with projection on head entity h. The distance function is defined as follows:

$$\mathbf{D}_{PP}(\mathbf{v}_h, \mathbf{v}'_h) = \left\|\mathbf{v}_h - \mathbf{v}'_h\right\|_1 \tag{3}$$

where $\|\mathbf{x}\|_1$ means the L1 norm function and $\mathbf{D}_{PP}$ means the point-to-point distance.

In summary, for IRI triplets, we compute the distance between two points in the embedding space, which ensures that the learned embeddings accurately represent the connections between different items in the KG.

TRT triplets. The *TRT* triplets reflect the connection between two tags. For such a triplet, its head and tail tags are both represented by a box, when using the relation for projection, it is necessary to consider not only the tag box center point translation but also modifying the tag box offset size. For example, for the triplet *(James Francis Cameron, is a citizen of, America)*, the objective is to ensure that the object box, mapped from *America* with relation, close to the box of *James Francis Cameron*. We utilize the center embedding and offset embedding of relation $\text{Cen}(\mathbf{b}_r), \text{Off}(\mathbf{b}_r)$ to achieve this goal as follows:

$$\text{Cen}(\mathbf{b}'_h) = \text{Cen}(\mathbf{b}_t) + \text{Cen}(\mathbf{b}_r) \tag{4}$$

$$\text{Off}(\mathbf{b}'_h) = \sigma(\text{Off}(\mathbf{b}_t)) + \text{Off}(\mathbf{b}_r) \tag{5}$$

where the σ is the $\texttt{ReLu}$ function in our paper since the box offset embedding of a tag $\text{Off}(\mathbf{b}_t)$ should be positive, while the element of $\text{Off}(\mathbf{b}_r)$ could be negative so it could be used to narrow $\sigma(\text{Off}(\mathbf{b}_t))$. The first operation 4 means changing the position of the box center point, while the second operation 5 means adjusting the size of the box.

The distance function of the TRT triplet should consider the similarity of the center and offset embedding at the same time, which is defined as follows:

$$\begin{aligned}\mathbf{D}_{BB}(\mathbf{b}_h, \mathbf{b}'_h) = &\left\|\text{Cen}(\mathbf{b}_h) - \text{Cen}(\mathbf{b}'_h)\right\|_1 + \\ &\left\|\sigma(\text{Off}(\mathbf{b}_h)) - \sigma(\text{Off}(\mathbf{b}'_h))\right\|_1\end{aligned} \tag{6}$$

where $\mathbf{D}_{BB}$ measures the distance between two box embeddings from the perspectives of center position deviation and size deviation.

In summary, for TRT triplets, we account for both the center and offset embeddings in calculating the similarity between boxes. This ensures that the learned embeddings accurately represent the connections between tags in the KG and allows for the consideration of the center and size of the boxes simultaneously.

IRT triplets. The *IRT* triplet contains an item and a tag, and the target is to locate the item point into the range of the concept box, which is generated by projecting the tag box using the relation box embedding. For a triplet (h, r, t) (h is an item and t is a tag), since even for the same tag like *James Francis Cameron*, different relation-tag pairs like *(is written by, James Francis Cameron)* and *(is directed by, James Francis Cameron)* are different concepts, the center embedding and offset embedding of relation $\text{Cen}(\mathbf{b}_r), \text{Off}(\mathbf{b}_r)$ are used to translate the tag box center $\text{Cen}(\mathbf{b}_t)$ and modify the offset $\text{Off}(\mathbf{b}_t)$, respectively, just as the operation for *TRT* triplets with Equation 4 and 5. For example, for the *IRT* triplet *(Avatar, is directed by, James Francis Cameron)*, the objective is to ensure that

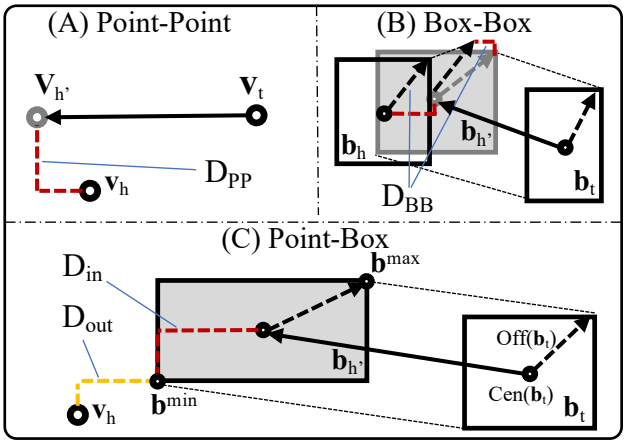

Figure 4: The geometric intuition of the different distances in 2-dimensional space. (A) The point-to-point distance for *IRI* triplets. (B) The box-to-box distance for *TRT* triplets. (C) The inside and outside distance for *IRT* triplets.

the concept box, mapped from *James Francis Cameron* with relation *is directed by*$^{-1}$, could contain the point of *Avatar*.

In order to measure whether the vector after projection is appropriate, we should define two types of distance between a point and a box, which are the outside distance and the inside distance:

$$\mathbf{D}_{PB}(\mathbf{v}_h, \mathbf{b}'_h) = \mathbf{D}_{out}(\mathbf{v}_h, \mathbf{b}'_h) + \mathbf{D}_{in}(\mathbf{v}_h, \mathbf{b}'_h) \tag{7}$$

where $\mathbf{D}_{PB}$ denotes point-to-box distance, while $\mathbf{D}_{out}(\mathbf{v}, \mathbf{b})$ and $\mathbf{D}_{in}(\mathbf{v}, \mathbf{b})$ is the outside distance and the inside distance, respectively. Note that the box is defined as an axis-aligned rectangle, so the two types of distance are calculated according to each dimension in the vector space as follows, which are defined as follows:

$$\mathbf{D}_{out}(\mathbf{v}, \mathbf{b}) = \left\|\texttt{Max}(\mathbf{v} - \mathbf{b}^{max}, \mathbf{0})\right\|_1 + \left\|\texttt{Min}(\mathbf{b}^{min} - \mathbf{v}, \mathbf{0})\right\|_1 \tag{8}$$

$$\mathbf{D}_{in}(\mathbf{v}, \mathbf{b}) = \left\|\text{Cen}(\mathbf{b}) - \texttt{Min}(\mathbf{b}^{max}, \texttt{Max}(\mathbf{b}^{min}, \mathbf{v}))\right\|_1 \tag{9}$$

where $\texttt{Max}$ and $\texttt{Min}$ are element-wise functions, which means selecting the larger or smaller element for each dimension. $\mathbf{b}^{max}$ and $\mathbf{b}^{min}$ are the points with the largest or smallest value in each dimension, which are calculated as follows:

$$\mathbf{b}^{max} = \text{Cen}(\mathbf{b}) + \sigma(\text{Off}(\mathbf{b})) \tag{10}$$

$$\mathbf{b}^{min} = \text{Cen}(\mathbf{b}) - \sigma(\text{Off}(\mathbf{b})) \tag{11}$$

According to the above definition, it can be seen that the point-to-box distance is used to measure the distance between a point and a box embedding calculating the distance between that point and the center point of the box and the closest point to the box border from each dimension.

In summary, for IRT triplets, $\mathbf{D}_{out}(\mathbf{v}_h, \mathbf{b}'_t)$ quantifies the proximity between point $\mathbf{v}_h$ and the nearest point within the box boundary, while $\mathbf{D}_{in}(\mathbf{v}_h, \mathbf{b}'_t)$ assesses the distance from the point to the box's center (provided the point resides inside the box along that dimension) or the offset magnitude (in the case that the point lies outside the box along that dimension).

Training and Optimization. In this step, we construct a loss function that considers positive and negative samples with different

sample weights to optimize the model. Specifically, for an *IRI* or *TRT* triplet to be predicted $(?, r, t)$ in the KG, n negative head entities will be randomly selected, which are not connected with t via relation r. For an *IRT* triplet (i, r, t), there are two approaches to generate a negative sample: replacing the item i or tag t. With the true triplets being positive samples, the goal of optimization is to minimize the distance of the positive samples and maximize the distance of negative samples. Therefore, the loss function is defined as:

$$\mathcal{L} = -w * (\log\theta(\gamma - \mathrm{D}^{pos}) - \frac{1}{n}\sum_{i=1}^{n}\log\theta(\gamma - \mathrm{D}^{neg})) \quad (12)$$

where θ is the `sigmoid` function, γ is a fixed scalar margin, n is the size of the negative sample, and w is the sample weight, which is set as the reciprocal of the number of correct answers, and it is used to balance the importance of each triplet in a training batch. D^{pos} and D^{neg} denote the distance of positive and negative triplets, which are calculated with Equation 3, 6, or 7 according to the triplet type. For each step, one type of triplet will be sampled, and the possibility is decided by its proportion in all types.

3.3 Box intersection

Beyond merely necessitating alignment between the item point and the concept box with respect to the triplets in the KG, it is also imperative for each item to reside within the intersection region formed by the amalgamation of its associated concepts box. Indeed, we posit that this composite concept encapsulates accurate and complex attributes of an item. While the Basic Pretraining step positions the item within each concept box, this intersection criterion is only implicitly fulfilled. Consequently, we carry out a secondary training phase with the objective of obtaining the intersection of a collection of boxes and situating the item point within the resultant intersecting box.

Let i represent the item, with its point embedding denoted as $\mathbf{v}$. Additionally, we obtain the tags $t_1, t_2, ..., t_n$ and relations $r_1, r_2, ..., r_n$ associated with item i based on the knowledge graph. The sets of tag and relation embeddings are represented as $\mathbf{b}_{t_1}, \mathbf{b}_{t_2}, ..., \mathbf{b}_{t_n}$ and $\mathbf{b}_{r_1}, \mathbf{b}_{r_2}, ..., \mathbf{b}_{r_n}$. Utilizing the projection operation in accordance with Equation 4 and 5 applied to the tag and relation embeddings, we obtain a final set of concept boxes $\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_n$. To ascertain the intersecting box using the embeddings, there are two approaches: one employing an attention neural network and the other a purely mathematical method. These are referred to as Attention Network Intersection and Max-Min Intersection, respectively.

Attention Network Intersection. The fundamental concept involves utilizing a neural network to determine the attention of each box center, as different boxes may have varying influences on the intersection region. The center embedding of $\mathbf{b}inter$ is defined as follows:

$$\mathrm{Cen}(\mathbf{b}_{inter}) = \sum_{i=1}^{n}\mathbf{a}_i \circ \mathrm{Cen}(\mathbf{b}_i) \quad (13)$$

where $\mathbf{a}_i \in \mathbb{R}^d$ represents the attention embedding and $\circ$ denotes element-wise multiplication. The core principle of Equation 13 is to allocate an attention value to each box for every dimension. a_i is generated using a `MLP` function:

$$\mathbf{a}_i = \frac{\exp(\mathrm{MLP}(\mathrm{Cen}(\mathbf{b}_i)))}{\sum_{j=1}^{n}\exp(\mathrm{MLP}(\mathrm{Cen}(\mathbf{b}_j)))} \quad (14)$$

For the offset embedding of the intersection box, it should be smaller than any offset in $\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_n$ in each dimension. Thus, we select the minimal element and generate a shrinking scale for each dimension with the offset embedding set $\mathrm{Off}(\mathbf{b}_1), \mathrm{Off}(\mathbf{b}_2), ..., \mathrm{Off}(\mathbf{b}_n)$:

$$\mathrm{Off}(\mathbf{b}_{inter}) = \mathrm{Min}(\sigma(\mathrm{Off}(\mathbf{b}_1)), ..., \sigma(\mathrm{Off}(\mathbf{b}_n))) \circ \mathbf{g} \quad (15)$$

where `Min` is an element-wise minimal function, and $\mathbf{g} \in \mathbb{R}^d$ represents the shrinking embedding, generated as follows:

$$\mathbf{g} = \theta(\mathrm{MLP}(\frac{1}{n}\sum_{i=1}^{n}\mathrm{MLP}(\mathrm{Off}(\mathbf{b}_i)))) \quad (16)$$

Max-Min Intersection. Since we are generating the intersection of multiple box embeddings, a more intuitive approach is to directly determine the intersection region across all the boxes in each dimension by selecting the maximal or minimal value from $\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_n$. Specifically, for the intersection box, we can obtain the max point $\mathbf{b}_{inter}^{max}$, defined in Equation 10, by selecting the minimal value from each box's max point in each dimension. Similarly, $\mathbf{b}_{inter}^{min}$ should choose the maximal value from the min points:

$$\mathbf{b}_{inter}^{max} = \mathtt{Min}(\sigma(\mathbf{b}_1^{max}), \sigma(\mathbf{b}_2^{max}), ..., \sigma(\mathbf{b}_n^{max})) \quad (17)$$

$$\mathbf{b}_{inter}^{min} = \mathtt{Max}(\sigma(\mathbf{b}_1^{min}), \sigma(\mathbf{b}_2^{min}), ..., \sigma(\mathbf{b}_n^{min})) \quad (18)$$

where the `Max` and `Min` are element-wise function just like Equation 8. According to Equation 10 and 11, with $\mathbf{b}_{inter}^{max}$ and $\mathbf{b}_{inter}^{min}$, we could calculate the $\mathrm{Cen}(\mathbf{b}_{inter})$ and $\mathrm{off}(\mathbf{b}_{inter})$ as follows:

$$\mathrm{Cen}(\mathbf{b}_{inter}) = (\mathbf{b}_{inter}^{max} + \mathbf{b}_{inter}^{min})/2 \quad (19)$$

$$\mathrm{Off}(\mathbf{b}_{inter}) = \sigma(\mathbf{b}_{inter}^{max} - \mathbf{b}_{inter}^{min})/2 \quad (20)$$

By the above method, we can obtain the embedding of the intersection region of multiple boxes using simple mathematical operations without neural networks.

Training and Optimization. In this step, given an item and its associated relation-tag pair, the intersection box embedding $\mathbf{b}_{inter} = (\mathrm{Cen}(\mathbf{b}_{inter}), \mathrm{Off}(\mathbf{b}_{inter}))$ can be obtained with the two methods mentioned above. Using the item embedding $\mathbf{v}$ and the intersection box embedding $\mathbf{b}_{inter} = (\mathrm{Cen}(\mathbf{b}_{inter}), \mathrm{Off}(\mathbf{b}_{inter}))$, the optimization goal is to position the item point within the box. The distance function is similar to Equation 7, which we employed for *IRT* triplets in the basic pretraining step. Negative samples are generated by replacing the item with other randomly selected items that are not related to these relation-tag pairs, and the loss function is identical to Equation 12. The sample weight w in the loss function is defined as $w = 1/(n+1)$, where n represents the number of concepts related to the item.

3.4 Interest Box Recommendation

The preceding two steps were trained only on the KG, with the objective of obtaining suitable representations for items, tags, and relations. Building on this foundation, we will utilize the pre-trained embedding for making predictions in the recommendation task. As previously discussed, a user's interests can be intricate and precise, making them challenging to represent with manually defined concepts like *a dress*. Instead, they may encompass more complex

concepts, such as *a red prom dress* (encompassing three aspects: red, prom, and dress), or even more intricate combinations of broader concepts. This amalgamation resembles the intersection of boxes. For example, a user could interact with an item i and an item j, and the combination concept of i and j is *red prom dress* and *prom highheels*, respectively. We could infer that the interest of this user may be interested in *prom outfit*.

In this step, each item's intersection box region is employed to generate the user's interest box. Since the influence of the same item may differ among users, a user-bias box intersection is conducted, which works in conjunction with the intersection results from the second training step. Ultimately, the user's interests are represented by a box embedding that integrates all items from the interaction history and is used to rank candidate items.

User-bias Intersection. For a user u assigned an embedding $\mathbf{u} \in \mathbb{R}^d$, the items in the intersection history form a set $i_1, i_2, ..., i_m$. In the second step, the intersection box of these items, denoted as $\mathbf{b}_{interI}$ ($interI$ represents intersection considering only the item), is generated without taking the user into account. To incorporate the user, we concatenate the user embedding $\mathbf{u}$ with each concept box center embedding $\text{Cen}(\mathbf{b})$ and offset embedding $\text{Off}(\mathbf{b})$, and use the combined embeddings to obtain the intersection box $\mathbf{b}_{interU}$ ($interU$ represents intersection considering the user) as follows:

$$\text{Cen}(\mathbf{b}_{interU}) = \sum_{i=1}^{n} \mathbf{c}_i \circ \text{Cen}(\mathbf{b}_i), \tag{21}$$

$$\text{Off}(\mathbf{b}_{interU}) = \sum_{i=1}^{n} \mathbf{d}_i \circ \text{Off}(\mathbf{b}_i) \tag{22}$$

$\mathbf{c}_i, \mathbf{d}_i \in \mathbb{R}^d$ are the attention embeddings generated as:

$$\mathbf{c}_i = \frac{\exp(\text{MLP}(\text{Cen}(\mathbf{b}_i), \mathbf{u})}{\sum_{j=1}^{m} \exp(\text{MLP}(\text{Cen}(\mathbf{b}_j), \mathbf{u})} \tag{23}$$

$$\mathbf{d}_i = \frac{\exp(\text{MLP}(\text{Off}(\mathbf{b}_i), \mathbf{u})}{\sum_{j=1}^{m} \exp(\text{MLP}(\text{Off}(\mathbf{b}_j), \mathbf{u})} \tag{24}$$

where $\text{MLP}(\cdot)$: $\mathbb{R}^{2d} \rightarrow \mathbb{R}^d$ transforms the $2d$-dimensional vector into a d-dimensional vector, which is distinct from the MLP in Equation 14 and 16.

Interest Box. With the two intersection boxes for each item, $\mathbf{b}_{interU}$ and $\mathbf{b}_{interI}$, the user's interest can be represented by the intersection box of their interacted items: $\mathbf{b}_{interU_1}, \mathbf{b}_{interU_2}, ..., \mathbf{b}_{interU_m}$ and $\mathbf{b}_{interI_1}, \mathbf{b}_{interI_2}, ..., \mathbf{b}_{interI_m}$. For each item, its final box embedding is the average of $\mathbf{b}_{interU}$ and $\mathbf{b}_{interI}$:

$$\text{Cen}(\mathbf{b}_{inter}) = (\text{Cen}(\mathbf{b}_{interI}) + \text{Cen}(\mathbf{b}_{interU}))/2 \tag{25}$$

$$\text{Off}(\mathbf{b}_{inter}) = (\text{Off}(\mathbf{b}_{interI}) + \text{Off}(\mathbf{b}_{interU}))/2 \tag{26}$$

The user u's interest considers all of the item boxes. With the item box embedding, we could form the user interest box embedding as follows:

$$\text{Cen}(\mathbf{b}_u) = \frac{1}{m} \sum_{k=1}^{m} \text{Cen}(\mathbf{b}_{inter\_k}) \tag{27}$$

$$\text{Off}(\mathbf{b}_u) = \frac{1}{m} \sum_{k=1}^{m} \text{Off}(\mathbf{b}_{inter\_k}) \tag{28}$$

This user interest box $\mathbf{b}_u$ is used to rank all items that have never been interacted with by the user according to the point-to-box distance.

Training and Optimization. For the recommendation task, the model generates the interest box for each user as described above. The optimization goal is to locate the interacted items' point embedding in the user's interest box while keeping the items that have not been engaged with the user outside of the box. Therefore, the point-to-box distance function $\mathbf{D}_{PB}(\mathbf{v}, \mathbf{b}_u)$ is adopted in this step. The negative samples are randomly selected from the items that are not in the interacted set $i_1, i_2, ..., i_m$. The loss function is the same as Equation 12, and the sample weight is defined as $w = 1/(m + \alpha)$, where m is the size of the interacted item set and α is a scale to balance the weight.

3.5 Model Prediction

After the three-stage training process, we obtain a point representation vector $\mathbf{v}_i$ for each item i and an interest box representation $\mathbf{b}_u$ for each user u encoded in the same embedding space. Now given a specific user-item pair (u, i) for recommendation, we can calculate an interest matching score between them using the proposed point-to-box distance function as follows:

$$\text{Score}(\mathbf{v}_i, \mathbf{b}_u) = \gamma - \text{D}_{PB}(\mathbf{v}_i, \mathbf{b}_u) \tag{29}$$

where γ is the scale used in Equation 12. The score is used to rank all the candidate items, and the higher the score, the higher the ranking.

4 EXPERIMENTS

In this segment, we embark on experiments utilizing four real-world datasets spanning diverse domains to thoroughly assess the efficacy of our proposed model. The empirical findings are geared towards addressing the following pivotal research inquiries:

- **RQ1**: How does our model compare against state-of-the-art recommendation baselines? This investigates the overall performance of integrating knowledge graph information to enhance recommendations.
- **RQ2**: What are the impacts of different components in our framework, including multi-stage training methodology, diverse data types, intersection mechanisms, etc? This provides an in-depth analysis of which design factors contribute the most to the performance gains.
- **RQ3**: Can we qualitatively validate if the box constraints successfully make items sharing common conceptual attributes have closer distributed representations as expected? This examines whether the knowledge graph integration achieves the goal of clustering semantically similar items in the embedding space.

Through comprehensive quantitative comparisons and carefully designed ablation studies towards answering the above key questions, we aim to validate the effectiveness of the knowledge graph-enhanced recommendation model proposed in this paper and provide insights into the best practices of integrating structured knowledge into recommender systems.

4.1 Experimental Settings

4.1.1 Datasets Description. We conduct our experiments on four diverse real-world datasets, extensively employed in prior research, to assess the comprehensive performance of InBox:

Last-FM[9] dataset is a valuable resource in recommendation system research, offering comprehensive user listening histories along with metadata on artists, albums, and songs.

Yelp2018[9] for business venue recommendation. Where local businesses like restaurants and bars are viewed as items.

Alibaba-iFashion[9] Alibaba-iFashion dataset is a fashion clothing dataset collected from the Alibaba online shopping system.

Amazon-Book[26] dataset is a collection of information on books sold on Amazon, including details such as title, author, price, ratings, reviews, and publication date.

As some cutting-edge models are based on GNN and require multi-hop neighbors of items in the KG, the datasets include two-hop neighbors to construct the knowledge graph. The overall statistics, encompassing user-item interaction history and the knowledge graph, are summarized in Table 1, and note that these datasets do not distinguish the interaction type. In accordance with our previous classification of KG triplet types in Section 2, we separately count the number of different triplet types and their proportions, which will prove beneficial for our experimental analysis.

Table 1: Statistics of the datasets used in our experiments. We report the basic statistics about the datasets, and notably, we also report the proportion of different types of triplets present in the KG.

Stas.	Last -FM	Yelp 2018	Alibaba-iFashion	Amazon -book
User-Item Interaction				
#Users	23,566	45,919	114,737	70,679
#Items	48,123	45,538	30,040	24,915
#Intersections	3,034,796	1,185,068	1,781,093	847,733
Knowledge Graph				
#Items	48,123	45,538	30,040	24,915
#Tags	58,266	90,961	59,156	88,572
#Relations	9	42	51	39
#IRI Triplets	3,284	0	0	2,985
#TRT Triplets	113,546	984,101	173,690	1,868,245
#IRT Triplets	347,737	869,603	105,465	686,516
IRI (%)	0.71%	0.00%	0.00%	0.12%
TRT (%)	24.44%	53.09%	62.22%	73.04%
IRT (%)	74.85%	46.91%	37.78%	26.84%

4.1.2 Evaluation Metrics. During the evaluation phase, we adopt the all-ranking strategy [15, 27] to ensure a fair comparison. Specifically, for each target user, all items not yet interacted with are considered negative, while their interacted items in the test set are deemed positive for inferring user preferences. All these items are ranked according to the matching scores produced by the model. To evaluate performance, we employ the established protocols [15]: recall@K and ndcg@K, with $K = 20$ as the default value.

The ndcg is calculated as:

$$\text{NDCG@K} = \frac{\sum_{i=1}^{k} \left(\frac{2^{y_i}-1}{\log_2(i+1)} \right)}{\sum_{i=1}^{k} \left(\frac{2^{y_{\pi_i}}}{\log_2(i+1)} \right)} \tag{30}$$

where k is the number of retrieved items considered, y_i is the number of retrieved items considered, and y_{π_i} is the relevance score of the item that would be at position i in an ideal ranking where all items are perfectly ranked according to their relevance.

The recall@K is

$$\text{recall@K} = \frac{\text{Number of relevant items in top K recommendations}}{\text{Total number of relevant items}} \tag{31}$$

4.1.3 Baselines. To demonstrate the effectiveness of our proposed model, we compare the overall performance of InBox with the state-of-the-art methods which include the KG-free method (MF), embedding-based methods (CKE, UGRec, and Hyper-Know), and propagation-based methods (LKGR, KGNN-LS, KGAT, CKAN, KGIN, and HAKG):

- **MF** [20] (Matrix factorization) is a benchmark factorization model, which only considers user-item interaction information without using KG.
- **CKE** [34] adopts TransR [16] to encode the items' semantic information and further incorporate it into the MF framework as item representation with the knowledge graph. KG relations are only used as the constraints in TransR training process.
- **UGRec** [35] is an embedding method that integrates the directed relations in KG and the undirected item-item co-occurrence relations simultaneously. While such undirected relations are unavailable for the datasets, the connectivities between items that are co-interacted by the same user are added as the co-occurrence relation.
- **Hyper-Know** [17] is an embedding-based method that embeds the knowledge graph in a hyperbolic space.
- **LKGR** [7] is a hyperbolic GNN-based method with Lorentz model. It uses different information propagation strategies to encode heterogeneous information.
- **KGNN-LS** [24] is a GNN-based model, which transforms KG into user-specific graphs and enriches item embeddings with GNN and label smoothness regularization.
- **KGAT** [26] is a GNN-based model, which iteratively applies an attentive message-passing scheme over the user-item-entity graph to generate user and item representations.
- **CKAN** [29] is based on KGNN-LS utilizing different neighborhood aggregation schemes on the user-item graph and KG respectively, to integrate the collaborative filtering representation space with the knowledge graph embedding.
- **KGIN** [27] is a GNN-based method to identify the latent intention of users with the interaction history, and performs GNN on the proposed user-intent-item-entity graph.
- **HAKG** [9] is a recently proposed GNN model that captures the underlying hierarchical structure of data in hyperbolic space.

Table 2: Overall Results of Our Model on Different Datasets and Comparison with Baseline Models. This table reports the NDCG and Recall metrics of different methods on various datasets. The best results are highlighted in bold and the second best results are underlined. The numbers in brackets indicate the relative improvements of our proposed method over the baseline models. The results reported for the baseline models are taken from prior published work.

	Last-FM		Yelp2018		Alibaba-iFashion		Amazon-book	
	recall	ndcg	recall	ndcg	recall	ndcg	recall	ndcg
MF	0.0724 (57.46%)	0.0617 (83.79%)	0.0627 (28.55%)	0.0413 (21.31%)	0.1095 (21.92%)	0.067 (26.57%)	0.1300 (34.77%)	0.0678 (41.15%)
CKE	0.0732 (55.74%)	0.0630 (80.00%)	0.0653 (23.43%)	0.0423 (18.44%)	0.1103 (21.03%)	0.0676 (25.44%)	0.1342 (30.55%)	0.0698 (37.11%)
UGRec	0.0730 (56.16%)	0.0624 (81.73%)	0.0651 (23.81%)	0.0419 (19.57%)	0.1006 (32.70%)	0.0621 (36.55%)	-	-
H-know	0.0948 (20.25%)	0.0812 (39.66%)	0.0685 (17.66%)	0.0447 (12.08%)	0.1057 (26.30%)	0.0648 (30.86%)	-	-
LKGR	0.0883 (29.11%)	0.0675 (68.00%)	0.0679 (18.70%)	0.0438 (14.38%)	0.1033 (29.24%)	0.0612 (38.56%)	-	-
KGNN-LS	0.0880 (29.55%)	0.0642 (76.64%)	0.0671 (20.12%)	0.0442 (13.35%)	0.1039 (28.49%)	0.0557 (52.24%)	0.1362 (28.63%)	0.056 (70.89%)
KGAT	0.0873 (30.58%)	0.0744 (52.42%)	0.0705 (14.33%)	0.0463 (8.21%)	0.1030 (29.61%)	0.0627 (35.25%)	0.1487 (17.82%)	0.0799 (19.77%)
CKAN	0.0812 (40.39%)	0.0660 (71.82%)	0.0646 (24.77%)	0.0441 (13.61%)	0.0970 (37.64%)	0.0509 (66.60%)	0.1442 (21.50%)	0.0698 (37.11%)
KGIN	0.0978 (16.56%)	0.0848 (33.73%)	0.0698 (15.47%)	0.0451 (11.09%)	0.1147 (16.39%)	0.0716 (18.44%)	0.1687 (3.85%)	0.0915 (4.59%)
HAKG	0.1008 (13.10%)	0.0931 (21.80%)	0.0778 (3.60%)	0.0501 (0.00%)	0.1319 (1.21%)	0.0848 (0.00%)	0.1642 (6.70%)	0.0907 (5.51%)
InBox	**0.1140**	**0.1134**	**0.0806**	**0.0501**	**0.1335**	**0.0848**	**0.1752**	**0.0957**

4.1.4 Parameter Settings. We implement the proposed InBox in Pytorch and train it using an RTX 3090 GPU. Our model is optimized with the Adam optimizer, with a fixed batch size of 256 for all three training steps and a negative sample size of 256 as well. The learning rate is initially set to 10^{-4}, transitioning to $2 * 10^{-5}$ and $4 * 10^{-6}$ when the training steps reach 50% and 75% of the maximum training steps, respectively. The number of training epochs is 100, 100, and 30 for the three training processes, respectively (except for the first training epoch for Amazon-Book, which is set to 8 due to the significantly larger number of triplets compared to other datasets). The embedding dimension is set to 512, and the margin γ in Equation 12 is set to 12. Furthermore, we employ an early stopping strategy when the recall@20 does not increase for two consecutive epochs. The baseline settings remain consistent with those in KGIN [27] and HAKG [9].

4.2 RQ1: Overall Performance

We present the overall performance on all datasets in Table 2, including recall@20 and ndcg@20, with the strongest baseline marked by an underline. In addition to the results, we also indicate the relative improvement of our model over each baseline. The baseline results of Last-FM, Yelp2018, and Alibaba-iFashion are taken from HAKG [9], while the results of Amazon-Book are from KGIN [27], except for HAKG's results on Amazon-Book, which are obtained using HAKG's open-source code. Our findings are as follows:

Compared to all the baselines, our model demonstrates superior performance across the four datasets, outperforming the strongest baseline with respect to recall@20 by 13.10%, 3.6%, 1.21%, and 3.85% in Last-FM, Yelp2018, Alibaba-iFashion, and Amazon-Book, respectively. This highlights the effectiveness of InBox. We attribute the improvements to (1) The box and point embeddings representing tags and items in the KG being more suitable for characterizing their connectivity in recommendation scenarios, resulting in a more reasonable distribution in vector space. In contrast, baselines that only use point embedding to model items and tags cannot clearly reflect this subordination. From another perspective, the relative positions of boxes and points also indicate the hierarchy of the entities in KG. In fact, HAKG, which considers the hierarchical relation of the KG, is generally closest to our method, yet still weaker, possibly because: (2) Concept combination more explicitly captures the complexity and diversity of user interests, making our model more expressive in capturing user interests than models that do not consider the connectivity of KG information and user interest in an explicit way.

Upon thoroughly analyzing the performance of InBox across four distinct datasets, we observe that the improvement on Last-FM is more pronounced than on the other three datasets. Referring to the triplet type ratio in Table 1, we believe this is due to the proportion of *IRT*-type triplets in Last-FM being the highest compared to other datasets. Besides utilizing *TRT* and *IRI* triplets in the first step, the model only employs *IRT* triplets in the subsequent two training processes. Simultaneously, we consider this an advantage of our model, as InBox has a lower demand for the completeness of the knowledge graph. *IRT* triplets are typically more crucial to train our model, so we do not require as much information about *IRI* and *TRT* triplets as other models, while GNN-based models depend on the quality of the entire KG information. This brings another advantage of our model, which is when the scale of the knowledge graph increases, assuming it is due to an increase in the number of nodes while keeping the edge density constant, the training cost for the model grows linearly with the scale of the model. If the increase is due to the edge density, then our model, which only requires one-hop information, clearly has an advantage over methods that integrate multi-hop neighbors.

In InBox, when obtaining the intersection box of an item, it can be considered as utilizing the one-hop neighbor information of the item. Compared to GNN-based models (i.e., KGAT and CKAN), although they claim to aggregate multi-hop neighbor information, their performance remains inferior to ours. Even when compared with other embedding-based baselines, such as CKE (focusing on one-hop neighbor information), the results are still not superior across all datasets. This may suggest that the importance of one-hop neighbors in KG plays a more crucial role than multi-hop nodes

in the recommendation scenario, and devoting more attention to one-hop neighbors could benefit the outcomes.

4.3 RQ2: Model Analysis

In this section, we conduct experiments to analyze the effectiveness of each training step and various components of the proposed model. The experimental results are summarized in Table 3, categorized by their corresponding training stages. For each experiment, we also show the relative performance improvement compared to the base model configuration to demonstrate the contribution of different components. Specifically, we incrementally validate the effects of the Data, different model functions, and training methods. Through these controlled experiments, we aim to provide an in-depth analysis of our multi-stage training methodology and reveal insights into which components are most critical for achieving strong performance. The analysis would guide us toward the most efficient model design by balancing model accuracy and training costs. Last but not least, we conduct a experiment of the hyper-parameter γ used in the loss function.

4.3.1 Analysis to the Basic Pretraining Step. We initially analyze the role the basic pretraining step plays in the training process, which encompasses two perspectives: (1) Is the entire training step crucial? (2) The primary results indicate that the *IRT* triplet may be more important; then, what will the results be if the other types of triplets are removed from the training stage? The first experiment eliminates this entire training step, i.e., training the model starting from the box intersection step, referred to as **w/o B** (without the Basic pretraining step). The second experiment excludes the *TRT* and *IRI* triplets during the first step, referred to as **only IRT**.

- The outcome achieved by removing the first step is marginally lower than the *base* results. We believe this is because the objectives of the first and second steps are similar. Although the learning targets differ, both steps attempt to obtain suitable point and box representations. Without the basic pretraining step, the box intersection step could still acquire appropriate embeddings.
- In the first step, removing *TRT* and *IRI* triplets has minimal impact on the results. Also, the subsequent two steps do not utilize *TRT* and *IRI* triplets. This illustrates that the *IRT* triplets in the knowledge graph may play a more direct and crucial role in the recommendation task than the other two types of triplets, and also offers a novel perspective on how to utilize KG rather than incorporating all multi-hop information. In real-world scenarios, the most readily available data for constructing a KG in a recommendation system is typically the attributes of the items. The interrelationships between items and the hierarchical relationships between attributes are often more difficult to obtain. From this perspective, our model has lower requirements for data, especially for data that is more difficult to obtain.

4.3.2 Analysis to the Box Intersection. For the second step of training, i.e., box intersection, we are also concerned about two issues: (1) What effect does removing the second step have on the results, termed **w/o I** (without box Intersection step)? (2) In the *base* experiment, we employ the attention network intersection method. What if we utilize the max-min intersection, termed **M-M I**.

- The *w/o I* result is also close to the *base* result, but compared with *w/o B*, it is slightly worse on all datasets. We attribute this difference to the fact that although the first and second steps can both obtain appropriate representations for entities and relations, the first step does not explicitly constrain that the item should be located in the intersection box, while the user interest box is based on the item intersection box. Positioning the item in each individual box, akin to the objective in the first step, may have similar requirements, but it is not direct enough. This also validates the rationality of concept combination and box intersection.
- Performing the intersection in a purely mathematical manner yields results close to the *base* experiment, which confirms the theoretical rationality of the box intersection operation. The neural network's improved adaptability to data noise and deviation may contribute to the superiority of *base* results over *M-M I*.

4.3.3 Analysis to the Third step. In the analysis of the third step, the recommendation step, we conduct three experiments: (1) Remove the first and second steps, and only train the model with the third step, termed **w/o B&I** (without Basic pretraining and Intersection); (2) When obtaining the user interest box, only use the intersection box acquired in the second step rather than adding the user-bias intersection box, termed **w/o userI**; (3) Only use user-bias intersection box in this step, termed **only userI**.

- Unlike the results of *w/o I* and *w/o B*, when the first and second steps are removed simultaneously, the performance significantly declines, demonstrating that the model does learn the positional connectivity of tags and items in the previous two steps. The results of *w/o B*, *w/o I*, and *w/o B&I* indicate that the first or second steps are not required simultaneously, but at least one of them is necessary.
- The results also demonstrate a slight but tangible absolute improvement in performance across various datasets, with a noteworthy relative improvement of over 4% in the Alibaba-iFashion dataset. The user bias introduced in the third training step is a critical component designed to inject a personalized element of randomness specific to each user, which was not addressed in the previous steps. This innovative addition is not merely an afterthought but a deliberate attempt to refine the recommendation process for individual users. Although the overall impact on system performance may be modest, as anticipated and consistent with our ablation study, the introduction of user bias is a significant step towards enhancing personalization. We look forward to exploring further ways to optimize this component and harness its full potential in future research.

4.3.4 Hyper-parameter Analysis. Our model training is divided into several training steps. Although the training objectives of each step are not the same, the training approach, especially in terms of the loss function, is similar. All loss functions involve the boundary

Table 3: Impact of each training step. For the basic pretraining step, we consider removing the whole step and some types of triplets, termed w/o B and only IRT; For the box intersection step, we consider removing the whole step and using a different intersection strategy, termed w/o I and M-M I; For the recommendation step, we consider removing the previous two steps, and the impact of user-bias intersection, termed w/o B&I, w/o userI, and only userI. The best results are highlighted in bold. The numbers in brackets indicate the relative improvements of our proposed method over the validation results.

	Last-FM		Yelp2018		Alibaba-iFashion		Amazon-book	
	recall	**ndcg**	**recall**	**ndcg**	**recall**	**ndcg**	**recall**	**ndcg**
w/o B	0.1092(4.40%)	0.1090(4.04%)	0.0796(1.26%)	0.0500(0.20%)	0.1276(4.62%)	0.0809(4.82%)	0.1733(1.10%)	0.0946(1.16%)
only IRT	0.1084(5.17%)	0.1077(5.29%)	0.0803(0.37%)	0.0501(0.00%)	0.1278(4.46%)	0.0809(4.82%)	0.1725(1.57%)	0.0943(1.48%)
w/o I	0.1069(6.64%)	0.1063(6.68%)	0.0779(3.47%)	0.0488(2.66%)	0.1274(4.79%)	0.0805(5.34%)	0.1665(5.23%)	0.0907(5.51%)
M-M I	0.1079(5.65%)	0.1072(5.78%)	0.0799(0.88%)	0.0500(0.20%)	0.1277(4.54%)	0.0809(4.82%)	0.1722(1.74%)	0.0943(1.48%)
w/o B&I	0.0363(214.05%)	0.0370(206.49%)	0.0602(33.89%)	0.0384(30.47%)	0.0684(95.18%)	0.0397(113.6%)	0.1059(65.44%)	0.0543(76.24%)
w/o userI	0.1114(2.33%)	0.1104(2.72%)	0.0795(1.38%)	0.0494(1.42%)	0.1280(4.30%)	0.0810(4.69%)	0.1737(0.86%)	0.0954(0.31%)
only userI	0.0621(83.57%)	0.0620(82.90%)	0.0738(9.21%)	0.0466(7.51%)	0.0920(45.11%)	0.0557(52.24%)	0.1479(18.46%)	0.0880(8.75%)
Base	**0.1140**	**0.1134**	**0.0806**	**0.0501**	**0.1335**	**0.0848**	**0.1752**	**0.0957**

hyperparameter γ, which serves to widen the gap between positive and negative sample triples. We conducted experiments to determine the impact of these parameters as Table 4 showes.

Table 4: The influence of γ in loss function.

	Last-FM	Yelp2018	Alibaba -iFashion	Amazon -book
$\gamma = 0$	0.0645	0.0324	0.0456	0.0559
$\gamma = 5$	0.0926	0.0392	0.0705	0.0813
$\gamma = 12$	0.1134	0.0501	0.0848	0.0957
$\gamma = 20$	0.1130	0.0494	0.0832	0.0946
$\gamma = 40$	0.1135	0.0499	0.0838	0.0955

As the table shows, in all datasets, when this parameter is small, it has a very noticeable impact on the results. This is because, under such circumstances, the training process cannot make a sufficient distinction between positive and negative samples, leading to poor outcomes. As this parameter increases, the performance gradually improves. However, after it reaches a certain level, further increasing the parameter has no effect on the results, because the distinction between positive and negative samples is already sufficiently large in the scale of the embedding representations, which aligns with the original intention of its design.

4.4 RQ3: Distribution Visualization and Item Distance Analysis

The final set of experiments, including the item distance measurements and visualizations, serves a critical purpose in our study. These experiments provide empirical evidence of the effectiveness of our model in capturing the inherent conceptual relationships between items as defined by the knowledge graph. By analyzing the distribution of items in the embedding space, we aim to demonstrate the model's ability to cluster semantically similar items, which is a key factor in enhancing recommendation relevance. Specifically, we randomly sample four relation-tag pairs, representing four concepts, and retrieve all items associated with these concept pairs from the Last-FM dataset, along with an equal number of randomly selected items that are not linked to the concepts.

To enable visualization, we reduce the high-dimensional item embeddings into two-dimensional points using principal component analysis (PCA). The distribution visualization is shown in Figure 5, with each subplot representing an example set.

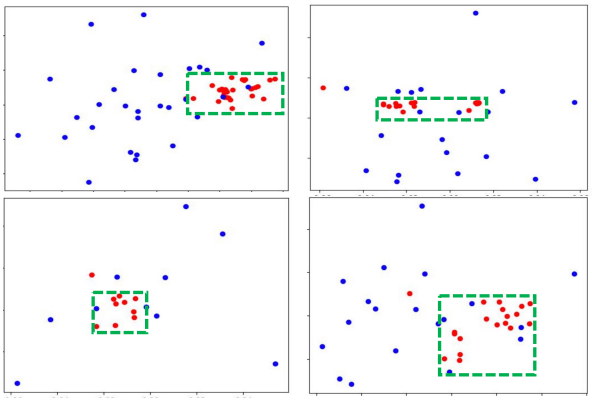

Figure 5: Four cases from *Last-FM*. The red points are the items connected to the relation-tag pair, and the blue points are randomly sampled items.

As depicted in Figure 5, items related to the concept (shown in red color) are clustered together, demonstrating that they share similar latent characteristics as defined by the relation-tag pair. On the other hand, the randomly selected unrelated items (in blue color) are dispersed arbitrarily across the 2D projected space without a clear pattern. The visualization experiment supplements these findings by providing a clear, graphical representation of how items associated with specific concepts are grouped together, while unrelated items are dispersed. This visual evidence strengthens our argument for the model's effectiveness in understanding and utilizing the KG's structure for improved recommendations.

Table 5: The distance of items with the same concept and the distance of items without shared concepts.

	set1	set2	set3	set4
D_related	0.403	0.504	0.349	0.504
D_random	0.857	0.861	0.795	0.878

Our item distance experiment further measures the average distance between items that share attributes within the same concept and compares it with the distance between randomly selected items as Table 5. This comparison reveals that items related to the same concept are significantly closer in the embedding space, validating the model's capacity to encode semantic similarities.

Together, these experiments highlight the interpretability of our model and its potential to offer insights into the recommendation process, which is vital for building trust and transparency in automated recommendation systems.

5 RELATED WORK

Existing KG-enhanced recommendation methods can be mainly categorized into several groups.

Path-based methods [6, 11, 13, 22, 25, 31, 33] extract paths that connect the target user and item nodes via KG entities, and with those paths, diffferent strategies like recurrent neural networks [28] or attention mechanism [11] could be applied to predict user interest. PER [33] also extracts meta-path based latent features to represent the connectivity between users and items. KPRN [28] generates path representations by composing the semantics of both entities and relations to infer the underlying high-order relation of a user-item interaction. To handle the large number of paths between two nodes, they define meta-path patterns to constrain the paths. RippleNet [22] automatically and iteratively extends a user's potential interests along links in KG. The method applying brute-force search easily leads to labor-intensive and time-consuming feature engineering. When using meta-path patterns to filter path instances, it relies on the domain -specific knowledge and human efforts, which limits the generalization to other scenario.

GNN-based methods [7, 9, 21, 24, 26, 27, 29] incorporate information from the neighbor nodes to update the representations of ego nodes; When such propagation is performed recursively, information from multi-hop nodes can be encoded in the representation finally. For example, KGAT [26] recursively performs propagation over the KG by combining graph convolution with an attention mechanism to obtain high-quality node representations. KGIN [27] captures user intents through the KG relations and disentangles user preference with intents for better interpretability. HAKG [9] models users and items as well as entities and relations in hyperbolic space, and design a hyperbolic aggregation scheme to gather relational contexts over KG. Despite the effectiveness, GNN-based models usually rely on the quality of KG, but in practical scenarios, KGs are usually sparse and noisy, which limits their performance.

Embedding-based [2, 5, 17, 23, 32, 34, 35]. methods generally incorporate KG embedding techniques (*e.g.*, TransE [4] and TransR [16]) to capture the KG structure and use additional KG loss to regularize the recommender model learning. CKE[34] uses TransR to jointly learn the latent representations as well as items' semantic representations from the knowledge graph. KTUP [5] adopts TransH [30] on the user-item intersection and knowledge graph. Hyper-Know [17] embeds knowledge graph in the hyperbolic space, which facilitates the learning of the hierarchical structure of KG and designs an adaptive regularization mechanism to regularize item representations.

In this work, we focus on modeling user interest in an appropriate way rather than a point. Our proposed model distinguishes itself from previous embedding-based models by learning box embedding [1, 8, 19] of user interests. Based on the idea that the conjunction of different concepts(*e.g.*, relation-tag pair in the KG can express user interests, we take the intersection of tags' box embedding as user interests. In this way, user preference is explicitly embedded as a box that captures the latent dependencies of user-tag-item relationships.

6 CONCLUSION

In this paper, we have presented a knowledge-aware recommendation model InBox, which introduces box representations to effectively encode the information from knowledge graphs. By modeling tags and relations as boxes geometrically, our approach enables more intuitive modeling of the connectivity and complex associations between item sets sharing common attributes. Moreover, we propose to represent each user's diverse interests as combinations of varying fundamental concept boxes via an intersection operation. The experiments on four real-world datasets have demonstrated the effectiveness of InBox, significantly outperforming state-of-the-art baselines. The in-depth quantitative analysis provides insights into the utility of different types of semantic knowledge relations, shedding light on what kind of information is most valuable for enhancing recommendations. The visualization also shows our model can produce more interpretable recommendations by uncovering the alignments between user preference and item attributes.

ACKNOWLEDGMENTS

This work is founded by National Natural Science Foundation of China (NSFC62306276/NSFCU23B2055/NSFCU19B2027/NSFC91846204), Zhejiang Provincial Natural Science Foundation of China (No. LQ23F020017), Ningbo Natural Science Foundation (2023J291), Yongjiang Talent Introduction Programme (2022A-238-G), Fundamental Research Funds for the Central Universities (226-2023-00138). This work was supported by AntGroup.

REFERENCES

[1] Ralph Abboud, İsmail İlkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. BoxE: A Box Embedding Model for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.).

[2] Qingyao Ai, Vahid Azizi, Xu Chen, and Yongfeng Zhang. 2018. Learning Heterogeneous Knowledge Base Embeddings for Explainable Recommendation. *Algorithms* 11, 9 (2018), 137. https://doi.org/10.3390/a11090137

[3] Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, Jason Tsong-Li Wang (Ed.). ACM, 1247–1250. https://doi.org/10.1145/1376616.1376746

[4] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 2787–2795.

[5] 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, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia (Eds.). ACM, 151–161. https://doi.org/10.1145/3308558.3313705

[6] Rose Catherine and William W. Cohen. 2016. Personalized Recommendations using Knowledge Graphs: A Probabilistic Logic Programming Approach. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, Shilad Sen, Werner Geyer, Jill Freyne, and Pablo Castells (Eds.). ACM, 325–332. https://doi.org/10.1145/2959100.2959131

[7] Yankai Chen, Menglin Yang, Yingxue Zhang, Mengchen Zhao, Ziqiao Meng, Jianye Hao, and Irwin King. 2022. Modeling Scale-free Graphs with Hyperbolic Geometry for Knowledge-aware Recommendation. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, K. Selcuk Candan, Huan Liu, Leman Akoglu, Xin Luna Dong, and Jiliang Tang (Eds.). ACM, 94–102. https://doi.org/10.1145/3488560.3498419

[8] Shib Dasgupta, Michael Boratko, Siddhartha Mishra, Shriya Atmakuri, Dhruvesh Patel, Xiang Li, and Andrew McCallum. 2022. Word2Box: Capturing Set-Theoretic Semantics of Words using Box Embeddings. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 2263–2276. https://doi.org/10.18653/v1/2022.acl-long.161

[9] Yuntao Du, Xinjun Zhu, Lu Chen, Baihua Zheng, and Yunjun Gao. 2022. HAKG: Hierarchy-Aware Knowledge Gated Network for Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 1390–1400. https://doi.org/10.1145/3477495.3531987

[10] Zhabiz Gharibshah, Xingquan Zhu, Arthur Hainline, and Michael Conway. 2020. Deep Learning for User Interest and Response Prediction in Online Display Advertising. *Data Sci. Eng.* 5, 1 (2020), 12–26. https://doi.org/10.1007/s41019-019-00115-y

[11] 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, KDD 2018, London, UK, August 19-23, 2018*, Yike Guo and Faisal Farooq (Eds.). ACM, 1531–1540. https://doi.org/10.1145/3219819.3219965

[12] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. IEEE Computer Society, 263–272. https://doi.org/10.1109/ICDM.2008.22

[13] Jiarui Jin, Jiarui Qin, Yuchen Fang, Kounianhua Du, Weinan Zhang, Yong Yu, Zheng Zhang, and Alexander J. Smola. 2020. An Efficient Neighborhood-based Interaction Model for Recommendation on Heterogeneous Graph. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 75–84. https://doi.org/10.1145/3394486.3403050

[14] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, Ying Li, Bing Liu, and Sunita Sarawagi (Eds.). ACM, 426–434. https://doi.org/10.1145/1401890.1401944

[15] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1748–1757. https://doi.org/10.1145/3394486.3403226

[16] 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 Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, Blai Bonet and Sven Koenig (Eds.). AAAI Press, 2181–2187.

[17] Chen Ma, Liheng Ma, Yingxue Zhang, Haolun Wu, Xue Liu, and Mark Coates. 2021. Knowledge-Enhanced Top-K Recommendation in Poincaré Ball. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 4285–4293.

[18] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[19] Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over Knowledge Graphs in Vector Space Using Box Embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

[20] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, Jeff A. Bilmes and Andrew Y. Ng (Eds.). AUAI Press, 452–461.

[21] 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 *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 51–60. https://doi.org/10.1145/3404835.3462912

[22] 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, CIKM 2018, Torino, Italy, October 22-26, 2018*, Alfredo Cuzzocrea, James Allan, Norman W. Paton, Divesh Srivastava, Rakesh Agrawal, Andrei Z. Broder, Mohammed J. Zaki, K. Selçuk Candan, Alexandros Labrinidis, Assaf Schuster, and Haixun Wang (Eds.). ACM, 417–426. https://doi.org/10.1145/3269206.3271739

[23] 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 on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis (Eds.). ACM, 1835–1844. https://doi.org/10.1145/3178876.3186175

[24] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. 2019. Knowledge-aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 968–977. https://doi.org/10.1145/3292500.3330836

[25] 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, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia (Eds.). ACM, 3307–3313. https://doi.org/10.1145/3308558.3313417

[26] 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, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 950–958. https://doi.org/10.1145/3292500.3330989

[27] 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 *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 878–887. https://doi.org/10.1145/3442381.3450133

[28] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable Reasoning over Knowledge Graphs for Recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019,*

The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019. AAAI Press, 5329–5336. https://doi.org/10.1609/aaai.v33i01.33015329

[29] Ze Wang, Guangyan Lin, Huobin Tan, Qinghong Chen, and Xiyang Liu. 2020. CKAN: Collaborative Knowledge-aware Attentive Network for Recommender Systems. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 219–228. https://doi.org/10.1145/3397271.3401141

[30] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, Carla E. Brodley and Peter Stone (Eds.). AAAI Press, 1112–1119.

[31] 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 *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 4486–4493.

[32] Xin Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon M. Jose. 2019. Relational Collaborative Filtering: Modeling Multiple Item Relations for Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer (Eds.). ACM, 125–134. https://doi.org/10.1145/3331184.3331188

[33] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: a heterogeneous information network approach. In *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*, Ben Carterette, Fernando Diaz, Carlos Castillo, and Donald Metzler (Eds.). ACM, 283–292. https://doi.org/10.1145/2556195.2556259

[34] 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, San Francisco, CA, USA, August 13-17, 2016*, Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (Eds.). ACM, 353–362. https://doi.org/10.1145/2939672.2939673

[35] Xinxiao Zhao, Zhiyong Cheng, Lei Zhu, Jiecai Zheng, and Xueqing Li. 2021. UGRec: Modeling Directed and Undirected Relations for Recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 193–202. https://doi.org/10.1145/3404835.3462835