Attribute Simulation for Item Embedding Enhancement in Multi-interest Recommendation

Yaokun Liu
College of Intelligence and Computing
Tianjin University
yaokunl@tju.edu.cn

Xiaowang Zhang
College of Intelligence and Computing
Tianjin University
xiaowangzhang@tju.edu.cn

Minghui Zou
College of Intelligence and Computing
Tianjin University
minghuizou@tju.edu.cn

Zhiyong Feng
College of Intelligence and Computing
Tianjin University
zyfeng@tju.edu.cn

ABSTRACT

Our research reveals that multi-interest recommendation models in the matching stage tend to exhibit an under-clustered item embedding space, which leads to a low discernibility between items and hampers item retrieval. This highlights the necessity for item embedding enhancement. However, item attributes, which serve as effective side information for enhancement, are either unavailable or incomplete in many public datasets due to the labor-intensive nature of manual annotation tasks. This dilemma raises two meaningful questions: 1. Can we bypass manual annotation and directly simulate complete attribute information from the interaction data? And 2. If feasible, how can we simulate attributes with high accuracy and low complexity in the matching stage?

In this paper, we first establish a theoretical feasibility that the item-attribute correlation matrix can be approximated through elementary transformations on the item co-occurrence matrix. Then, based on further formula derivation, we propose a simple yet effective module, **SimEmb** (Item Embedding Enhancement via Simulated Attribute), in the multi-interest recommendation of the matching stage. By simulating attributes with the co-occurrence matrix, SimEmb discards the ID-based item embedding and employs the attribute-weighted summation for item embedding enhancement. Comprehensive experiments on four benchmark datasets demonstrate that our approach notably enhances the clustering of item embedding and significantly outperforms SOTA models with an average improvement of 25.59% on Recall@20.

CCS CONCEPTS

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

KEYWORDS

Recommender Systems, Multi-interest Learning, Attribute Simulation, Item Embedding Enhancement

WSDM '24, March 4–8, 2024, Merida, Mexico

ACM ISBN 979-8-4007-0371-3/24/03…$15.00
https://doi.org/10.1145/3616855.3635841

ACM Reference Format:
Yaokun Liu, Xiaowang Zhang, Minghui Zou, and Zhiyong Feng. 2024. Attribute Simulation for Item Embedding Enhancement in Multi-interest Recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM '24), March 4–8, 2024, Merida, Mexico.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3616855.3635841

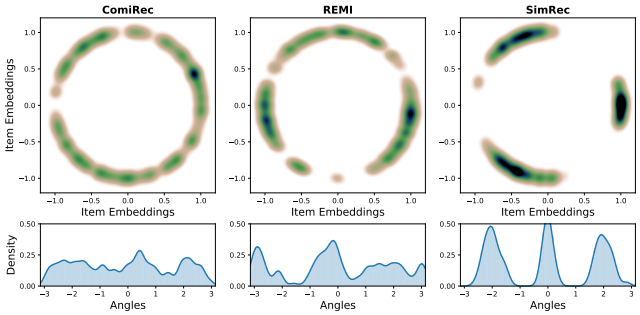

Figure 1: Item embedding distributions and density estimation curves of 600 items randomly sampled from Retail-Rocket, where darker colors indicate denser point clustering in this area. More details can be found in Section 5.7.

1 INTRODUCTION

To deliver personalized filtering in online platforms' vast item pools, industrial recommendation systems [11, 30] commonly follow a two-stage process: matching and ranking. The matching stage extracts a small subset of items from the extensive corpus with lightweight models to ensure low computational costs. Subsequently, the ranking stage reranks the retrieved items with more sophisticated models. This paper focuses on the matching stage, which serves as a crucial foundation for the recommender system.

Recently, a rise of multi-interest learning-based methods has been witnessed in the sequential recommendation of the matching stage [13, 49]. The multi-interest recommendation extracts multiple interest vectors for each user based on their interaction sequences, effectively alleviating the information loss of using a single user embedding and gaining notable improvement in matching performance. MIND [25] pioneers using the capsule network with dynamic routing [35] to capture multiple interests. Comirec [6] introduces multi-head self-attention mechanisms and explores the balance between diversity and accuracy. Subsequent work [7, 8, 27, 46] further improves from different aspects such as periodicity, user profile, and negative sampling.

Despite their effectiveness, our investigation reveals a clustering deficiency problem in the item embedding space of existing multi-interest models. As shown in Figure 1, it is evident that the item embeddings of classic ComiRec and SOTA REMI models are uniformly distributed across the entire circle (in stark contrast to our proposed SimRec). This high similarity between embeddings of different items hampers item-matching efficiency, suggesting the need for enhancing item embeddings.

An effective way to enhance item embeddings is leveraging item attributes, which provide rich semantic information. However, most multi-interest methods do not consider the item attribute. This is reasonable because of the time-consuming and labor-intensive nature of annotating item attributes, making item attributes unavailable or incomplete in many public datasets. It naturally leads to a meaningful question: *Can we bypass the need for manual annotation of item attributes and, instead, simulate the attribute-item correlation from interaction data to achieve approximate effects?*

To answer the question, we find that the item co-occurrence information inherently contains item-attribute correlations. As shown in Figure 2, item co-occurrence within user sequences strongly implies specific similarities, indicating shared attributes among items. Exploring this implicit containment relationship, our theoretical analysis in Section 3 shows that applying elementary transformations to the item co-occurrence matrix can approximate the actual item-attribute correlation matrix.

While we establish the theoretical feasibility of simulating attributes, translating this into practical application within the matching stage presents challenges. Determining the correct transformation steps and managing the potential high complexity due to the co-occurrence matrix's high dimensionality is non-trivial. Therefore, a subsequent query arises: *Is there an implementation method to achieve both high simulation accuracy and low complexity?*

In this paper, we affirmatively address the question by deriving a formula in Section 4.2. This formula reveals that attribute simulation and attribute-weighted summation can be simplified into an intriguing equation: the multiplication of the co-occurrence matrix by the attribute embedding matrix yields enhanced item embeddings. Based on this finding, we propose a simple yet effective module named **SimEmb** (Item Embedding Enhancement via Simulated Attribute). Specifically, we first construct a normalized item co-occurrence matrix from user sequences. Then, we discard the existing item ID-based embedding layer and perform attribute-weighted summation based on the co-occurrence matrix to obtain the enhanced item embeddings. Finally, to reduce the training complexity further, we propose a process optimization of SimEmb. The contributions of this paper can be summarized as follows:

- We provide a theoretical feasibility that the item co-occurrence matrix can approximate the real item-attribute correlation matrix through elementary transformations.
- We propose a simple yet effective item embedding module that enhances the item embedding through attribution simulation with the item co-occurrence matrix. It can be an ideal alternative to the labor-intensive attribute annotating task.
- We conduct comprehensive experiments on four benchmark datasets, and the results demonstrate SimEmb significantly improves the effectiveness of various multi-interest models, with an average increase of 25.59% on Recall@20.

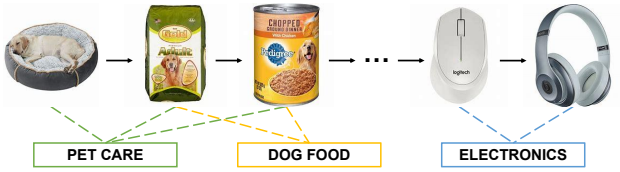

Figure 2: An example of a user sequence and shared attributes between items.

2 PRELIMINARIES

2.1 Problem Formulation

Let $\mathcal{U}$ denotes the set of users and $\mathcal{I}$ denotes the corpus of items. For a given user $u \in \mathcal{U}$, we have a historical interaction sequence $\mathcal{S}^u = \{i_1^u, i_2^u, \ldots, i_L^u\}$ ordered chronologically, where $i_l^u \in \mathcal{I}$ denotes the l-th item the user interacted with and L denotes the maximum length of the user behavior sequence. In the matching stage, the goal of multi-interest recommendation is to retrieve a subset of items from $\mathcal{I}$ that user is likely to interact with, based on multiple interests vector extracted from $\mathcal{S}^u$.

2.2 Multi-interest Framework

In this section, we provide an overview of the existing multi-interest recommendation framework.

2.2.1 Embedding Layer. Let $\mathbf{E}_{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}| \times d}$ denotes an item embedding matrix that maps all one-hot item IDs into d-dimensional dense embedding vectors. Through embedding look-up, the input user interaction sequence $\mathcal{S}^u$, which comprises a series of item IDs, is converted into a matrix $\mathbf{H}$:

$$\mathbf{H} = \text{Embedding}(\mathcal{S}^u; \mathbf{E}_{\mathcal{I}}) = \left[\mathbf{e}_1^u, \mathbf{e}_2^u, \ldots, \mathbf{e}_L^u\right] \in \mathbb{R}^{L \times d} \tag{1}$$

where $\mathbf{e}_l^u \in \mathbb{R}^{1 \times d}$ is the embedding of the l-th item i_l^u.

2.2.2 Multi-interest Extraction Module. The aim of this module is to calculate the item-to-interest weight matrix $\mathbf{W}$ based on the embedded user sequences, and then obtain the multi-interest matrix $\mathbf{V}_u$ for the user u:

$$\mathbf{W} = \mathcal{F}(\mathbf{H}) \in \mathbb{R}^{K \times L} \tag{2}$$

$$\mathbf{V}_u = \mathbf{W}\mathbf{H} \in \mathbb{R}^{K \times d} \tag{3}$$

where $\mathcal{F}$ represents the function for obtaining the weight matrix, and K is the number of interest vectors.

2.2.3 Training and Serving. In the training phase, given the target item i^+ and $\mathbf{V}_u$, we first utilize the argmax operator to select a interest vector which most relevant to the target item:

$$\mathbf{v}_u = \mathbf{V}_u \left[\text{argmax}(\mathbf{V}_u \mathbf{e}_{i^+}^\top), :\right] \in \mathbb{R}^{1 \times d} \tag{4}$$

where $\mathbf{e}_{i^+}$ is embedding of the i^+. Based on the selected user interest $\mathbf{v}_u$, the probability of user-item interaction $P(i^+ \mid u)$ can be calculated as follows:

$$P(i^+ \mid u) = \frac{\exp(\mathbf{v}_u \mathbf{e}_{i^+}^\top)}{\exp(\mathbf{v}_u \mathbf{e}_{i^+}^\top) + \sum_{i^- \in \mathcal{N}} \exp(\mathbf{v}_u \mathbf{e}_{i^-}^\top)} \tag{5}$$

To alleviate computational complexity resulting from the large number of items, the sampled softmax [19] is employed here, with $\mathcal{N}$ denoting the set of randomly sampled negative samples shared

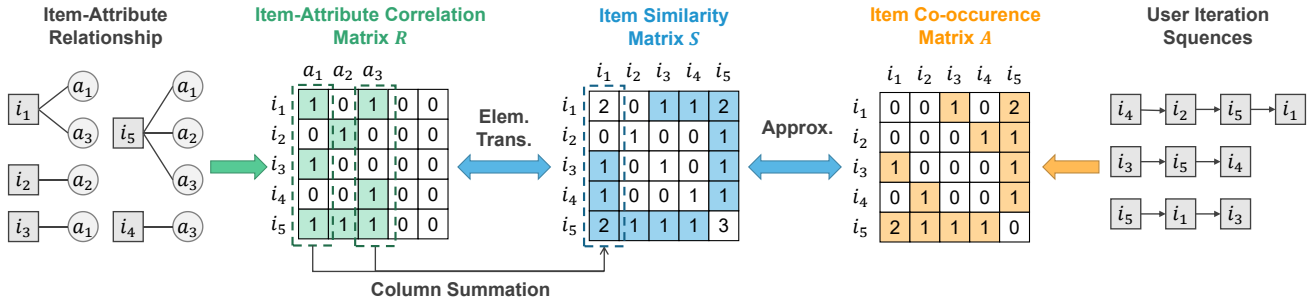

Figure 3: The relationship between item-attribute correlation matrix and Item co-occurence matrix. For ease of illustration, we consider a specific scenario with only five items and three attributes.

in a batch. The formulated loss function aims to maximize $P(i^+ \mid u)$.The loss function is formulated as follows to maximize $P(i^+ \mid u)$:

$$\mathcal{L}(\theta) = \sum_{u \in \mathcal{U}} -\log P(i^+ \mid u) \tag{6}$$

For the online serving phase, we adopt the fast nearest neighbors algorithm (*e.g.*, Faiss [20]) to retrieve the top-N most relevant items for each interest vector from the large-scale item corpus. Then we calculate the relevance scores $f(u, i)$ between the user and these K×N candidate items:

$$f(u, i) = \max_{1 \le k \le K} (\mathbf{v}_u^{(k)} \mathbf{e}_i^\top) \tag{7}$$

where $\mathbf{v}_u^{(k)}$ is the k-th interest of user u. Finally, we use the $f(u, i)$ to reselect the top-N items as the final recommendation results.

3 INVESTIGATION OF ATTRIBUTE SIMULATION FROM INTERACTION DATA

Within the multi-interest recommendation framework, there is a frequent need to calculate item similarity with simple functions like dot product on item embeddings. Therefore, a well-clustered item embedding space plays a crucial role in improving the matching performance of the model. Although attribute has proven to be effective side information for item embedding enhancement, the laborious task of annotating results in their unavailability or incompleteness in many public datasets. To address this challenge, we theoretically explore the potential of simulating item-attribute correlations from user-item interaction in this section.

Assume we have a complete attribute set $\mathcal{A}$. By using one-hot ID encoding from 1 to $|\mathcal{I}|$ for items and similarly from 1 to $|\mathcal{A}|$ for attributes, the item $i_i \in \mathcal{I}$ denotes the item with ID i, and $a_j \in \mathcal{A}$ denotes the attribute with ID j. As shown in Figure 3, we can depict the relationship between items and attributes as a binary item-attribute correlation matrix $\mathbf{R} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{A}|}$, which represents items in rows and attributes in columns. The element $r_{ij} = 1$ signifies item i_i possesses attribute a_j, while $r_{ij} = 0$ indicates the opposite.

For the item i_i, we can obtain the ID set of its attributes as:

$$C_i = \{ j \mid r_{i,j} \neq 0, 1 \le j \le |\mathcal{A}| \} \tag{8}$$

Consider a matrix $\mathbf{S} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$, where the i-th column is equal to the sum of the column vectors of matrix $\mathbf{R}$ with indices in C_i:

$$\mathbf{S}[:, i] = \sum_{j \in C_i} \mathbf{R}[:, j] \in \mathbb{R}^{|\mathcal{I}| \times 1} \tag{9}$$

Upon applying Equation (9) to all columns of matrix $\mathbf{S}$, it is evident that $\mathbf{S}$ is derived via elementary column transformations and matrix extension on $\mathbf{R}$, which can formulate as:

$$\mathbf{S} = [\mathbf{R} : \mathbf{O}]\, \mathbf{P} \tag{10}$$

where $[:]$ denotes the horizontal concatenation of two matrices. $\mathbf{O} \in \mathbb{R}^{|\mathcal{I}| \times (|\mathcal{I}| - |\mathcal{A}|)}$ is a zero matrix, and $\mathbf{P} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$ is an elementary matrix.

The off-diagonal elements in $\mathbf{S}$ depict the similarity between items. For example, as shown in Figure 3, the element $s_{15} = 2$ indicates a certain degree of similarity between item i_1 and i_5 with two shared attributes, while $s_{12} = 0$ indicates no similarity between item i_1 and i_2 due to the absence of shared attributes. Hence, matrix $\mathbf{S}$ also serves as an item similarity matrix.

The item similarity naturally prompts us with the co-occurrence of items in the user interaction sequence. As the consecutive appearance of two items suggests their potential similarity and a higher co-occurrence frequency indicates a stronger similarity, the item co-occurrence matrix can also reflect the similarity between items to some extent. Suppose $\mathbf{A} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$ is an item co-occurrence matrix, where element a_{ij} represents the number of times item i_i and item i_j co-occur. After normalizing, $\mathbf{A}$ can be viewd as a statistical approximation of the item similarity matrix $\mathbf{S}$. Furthermore, the following formula can be derived:

$$[\mathbf{R} : \mathbf{O}] = \mathbf{S}\mathbf{P}^{-1} \approx \mathbf{A}\mathbf{P}^{-1} \tag{11}$$

where $\mathbf{P}^{-1} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$ is the inverse of elementary matrix $\mathbf{P}$. This formula demonstrates that by applying reverse elementary column transformations to matrix $\mathbf{A}$, we can obtain an approximation of the item-attribute correlation matrix $\mathbf{R}$ without manual annotation.

4 METHOD

The findings in Section 3 provide us with an inspiring theoretical feasibility of attribute simulation. However, applying it in the multi-interest learning of the matching stage is not trivial due to the need for both high simulation accuracy and low complexity. In this section, we introduce our solution and propose a simple yet effective item embedding enhancement module.

4.1 Construction of Item Co-occurrence Matrix

Before training, we first create an item co-occurrence matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$ initializing as a zero matrix. By iterating through all user sequences in the training set, we execute the following operation

on the matrix element a_{ij} when item i_i and item i_j co-occur:

$$a_{ij} = \begin{cases} a_{ij}, & \text{if } T - d_{ij} < 0 \\ a_{ij} + T - d_{ij}, & \text{otherwise} \end{cases} \tag{12}$$

where d_{ij} is the step interval between i_i and i_j and T is the threshold for the step interval. As the co-occurrence matrix is symmetric, the same operation is also applied to a_{ji}. Here, we assign higher values to pairs of closer items, emphasizing the higher likelihood of similarity between them. Moreover, we set the diagonal elements of matrix A to 1 and proceed with row normalization.

4.2 Item Embedding Enhancement via Simulated Attribute

To tackle the problem of inadequate item embeddings, our goal is to integrate attribute side information for enhancement.

Ideally, we would have complete attribute information for all items in the item-attribute correlation matrix $\mathbf{R}$. Let $\mathbf{E}_{\mathcal{A}} \in \mathbb{R}^{|\mathcal{A}| \times d}$ denotes the embedding matrix of attributes and $\mathbf{e}_{a_j} \in \mathbb{R}^{1 \times d}$ denotes the embedding of attribute a_j obtained by querying $\mathbf{E}_{\mathcal{A}}$. We can derive the enhanced item embedding $\mathbf{e}_{i_i}$ of item i_i as:

$$\mathbf{e}_{i_i} = \sum_{j=1}^{|\mathcal{A}|} r_{ij} \mathbf{e}_{a_j} \in \mathbb{R}^{1 \times d} \tag{13}$$

where r_{ij} is the element in the item-attribute correlation matrix $\mathbf{R}$. It is important to mention that this formula can also add item ID-based embedding. However, in an ideal situation where all attributes are known, just as there are no two identical leaves in the world, there would not be two items with exactly all the same attributes. Therefore, unlike the coarse-grained attributes obtained through manual labeling, when attributes are fine-grained and comprehensive, relying solely on item attributes suffices for item representation, and there will not be duplicate item embeddings. Consequently, adding item ID-based item embeddings is redundant.

We can further implement the Equation (13) in matrix form as:

$$\mathbf{E}_{\mathcal{I}} = \mathbf{R}\mathbf{E}_{\mathcal{A}} = [\mathbf{R} : \mathbf{O}]\, \mathbf{E}'_{\mathcal{A}} \tag{14}$$

where $\mathbf{E}_{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}| \times d}$ denotes the enhanced item embedding matrix, and $\mathbf{E}'_{\mathcal{A}} \in \mathbb{R}^{|\mathcal{I}| \times d}$ is obtained by vertically expanding matrix $\mathbf{E}_{\mathcal{A}}$. Since $\mathbf{R}$ is unavailable, following Equation (11), we can further substitute with the co-occurrence matrix $\mathbf{A}$ as:

$$\mathbf{E}_{\mathcal{I}} = \mathbf{A}\mathbf{P}^{-1}\mathbf{E}'_{\mathcal{A}} \tag{15}$$

Since the elementary matrix $\mathbf{P}^{-1}$ is unidentified, we can treat it as a parameter matrix. However, as $\mathbf{P}^{-1}$ is a square matrix with a high dimension of $\mathcal{I}$, it will bring a notable increase in the model's parameter count, and performing multiplications between such sizable matrices also results in unbearable computational complexity.

Facing these challenges, it is necessary to further simplify the formula. According to the associative law of matrix multiplication, we can rewrite Equation (15) as follows:

$$\mathbf{E}_{\mathcal{I}} = \mathbf{A}\left(\mathbf{P}^{-1}\mathbf{E}'_{\mathcal{A}}\right) \tag{16}$$

We can observe that $\mathbf{P}^{-1}\mathbf{E}'_{\mathcal{A}}$ actually conducts row elementary transformations on the embedding matrix $\mathbf{E}'_{\mathcal{A}}$. Hence, we can merge the process by directly introducing a unified parameter matrix $\widetilde{\mathbf{E}} \in \mathbb{R}^{|\mathcal{I}| \times d}$ to represent the embedding matrix $\mathbf{E}'_{\mathcal{A}}$ after elementary transformations. As a result, Equation (16) can simplify as:

$$\mathbf{E}_{\mathcal{I}} = \mathbf{A}\widetilde{\mathbf{E}} \tag{17}$$

Interestingly, we can observe a formal resemblance between Equation (14) and (17), which suggests that we can regard the item co-occurrence matrix $\mathbf{A}$ as a surrogate item-attribute co-occurrence matrix $\mathbf{R}$, and $\widetilde{\mathbf{E}}$ takes on the role of a dimension-aligned attribute embedding matrix. Through their multiplication, we can concurrently accomplish attribute simulation and attribute-weighted summation, resulting in an enhanced item embedding matrix.

In summary, we replace the item ID-based embedding layer with our proposed SimEmb module, which follows this overall workflow:

1) Perform a multiplication between the item co-occurrence matrix and the attribute embedding matrix to obtain an enhanced item embedding matrix $\mathbf{E}_{\mathcal{I}}$, as described in Equation (17).

2) Embed the input user sequence $\mathcal{S}^u$ by performing embedding-lookup on $\mathbf{E}_{\mathcal{I}}$, as described in Equation (1).

Table 1: Dataset Statistics.

Datasets	#Users	#Items	#Interactions	Density
Beauty	22,363	12,101	198,502	0.0734%
Books	603,668	367,982	8,898,041	0.0040%
RetailRocket	37,175	25,310	440,084	0.0468%
Yelp	19,242	14,142	201,237	0.0740%

4.3 Complexity Analysis and Process Optimization

In this section, we analyze the additional time complexity caused by replacing the item ID-based embedding layer with the SimEmb module and refine its processing for optimization.

In the training phase, the additional time complexity primarily originates from Equation (17). Considering that the item co-occurrence matrix $\mathbf{A}$ takes the sparse matrix form and assuming its density is ρ (detailed values are presented in Table 1), the resultant time complexity is $O(\rho|\mathcal{I}|^2 d)$. This complexity is impractical due to the huge $|\mathcal{I}|$ in the matching stage. Therefore, we further optimize the process of SimEmb as follows:

1) Perform the embedding-lookup on $\mathbf{A}$ to yield the correlation matrix $\mathbf{C}$ between items in the user sequence $\mathcal{S}^u$ and their attributes:

$$\mathbf{C} = \text{Embedding}\left(\mathcal{S}^u; \mathbf{A}\right) \in \mathbb{R}^{L \times |\mathcal{I}|} \tag{18}$$

2) Derive the enhanced item embeddings via weighted summation on attribute embeddings:

$$\mathbf{H} = \mathbf{C}\widetilde{\mathbf{E}} = \left[\mathbf{e}_1^u, \mathbf{e}_2^u, \ldots, \mathbf{e}_L^u\right] \in \mathbb{R}^{L \times d} \tag{19}$$

Following the optimized process, the additional time complexity primarily originates from Equation (19) and can be notably reduced to $O(\rho L|\mathcal{I}|d)$, which is demonstrated to be totally acceptable by experiments in Section 5.6.

In the serving phase, it's important to highlight that there is no need to operate attribute-weighted summation as we can preserve the learned enhanced item embedding matrix by operating Equation (17) once and directly conduct embedding-lookup on it. Therefore, there is no additional time complexity introduced.

Table 2: Model comparison. Bold: best performance. Underlined: suboptimal performance.

Datasets	Metric	PopRec	Y-DNN	GRU4Rec	MIND	ComiRec	PIMIRec	Re4	REMI	SimRec	Improv.
Beauty	R@20	0.0236	0.0643	0.0480	0.0822	0.0650	0.0731	0.0741	0.0779	**0.1139**	+46.21%
	R@50	0.0473	0.1025	0.0768	0.1259	0.1102	0.1101	0.1142	0.1259	**0.1810**	+43.76%
	ND@20	0.0173	0.0526	0.0414	0.0650	0.0415	0.0477	0.0539	0.0537	**0.0758**	+40.63%
	ND@50	0.0260	0.0642	0.0502	0.0769	0.0554	0.0631	0.0665	0.0687	**0.0927**	+34.93%
	HR@20	0.0411	0.1135	0.0845	0.1390	0.1100	0.1305	0.1229	0.1346	**0.1873**	+39.15%
	HR@50	0.0845	0.1748	0.1310	0.1989	0.1775	0.2070	0.1873	0.2088	**0.2758**	+32.09%
Books	R@20	0.0144	0.0380	0.0295	0.0636	0.0525	0.0723	0.0602	0.0768	**0.0937**	+22.01%
	R@50	0.0246	0.0620	0.0472	0.0937	0.0819	0.1107	0.0994	0.1123	**0.1408**	+25.38%
	ND@20	0.0129	0.0323	0.0291	0.0639	0.0366	0.0510	0.0552	0.0636	**0.0715**	+12.42%
	ND@50	0.0171	0.0419	0.0365	0.0746	0.0484	0.0652	0.0700	0.0765	**0.0877**	+14.64%
	HR@20	0.0318	0.0834	0.0642	0.1323	0.1088	0.1449	0.1305	0.1532	**0.1799**	+17.43%
	HR@50	0.0535	0.1323	0.1026	0.1897	0.1670	0.2144	0.2067	0.2189	**0.2607**	+19.10%
Retail Rocket	R@20	0.0190	0.3096	0.2762	0.3797	0.4133	0.4179	0.4631	0.4537	**0.4859**	+7.10%
	R@50	0.0319	0.3896	0.3318	0.4753	0.4959	0.5110	0.5385	0.5483	**0.6012**	+9.65%
	ND@20	0.0161	0.2597	0.2637	0.3281	0.3709	0.3643	0.3580	**0.3877**	0.3812	-0.17%
	ND@50	0.0202	0.2760	0.2732	0.3406	0.3829	0.3757	0.3677	**0.3990**	0.3946	-0.11%
	HR@20	0.0379	0.4519	0.4139	0.5519	0.5906	0.5912	0.6226	0.6283	**0.6600**	+5.05%
	HR@50	0.0605	0.5468	0.4806	0.6455	0.6772	0.6807	0.6934	0.7160	**0.7577**	+5.82%
Yelp	R@20	0.0257	0.0549	0.0244	0.0624	0.0625	0.0842	0.0598	0.0725	**0.0921**	+27.03%
	R@50	0.0474	0.1034	0.0470	0.1373	0.1282	0.1487	0.1183	0.1328	**0.1881**	+41.64%
	ND@20	0.0194	0.0462	0.0212	0.0479	0.0547	0.0644	0.0489	0.0638	**0.0679**	+5.43%
	ND@50	0.0280	0.0650	0.0300	0.0764	0.0787	0.0861	0.0695	0.0852	**0.0984**	+15.49%
	HR@20	0.0468	0.1179	0.0468	0.1319	0.1340	0.1668	0.1210	0.1616	**0.1839**	+13.80%
	HR@50	0.0914	0.2135	0.0919	0.2655	0.2556	0.2784	0.2270	0.2732	**0.3345**	+22.44%

5 EXPERIMENTS

5.1 Experimental Setting

5.1.1 Datasets. We conduct experiments on four benchmark datasets:

- **Amazon Beauty**[1] and **Books**[1] [31] contain user reviews from Amazon.com. We select two widely used categories.
- **RetailRocket**[2] is a real-world e-commerce dataset widely used for research in personalized recommendation.
- **Yelp**[3] is a dataset of user-generated reviews and ratings for businesses collected from the Yelp platform. We only keep the records from January 1st, 2019, to December 31st, 2019.

We preprocess the dataset according to [6], treating all interactions as implicit feedback. Users or items with fewer than five occurrences are removed. The maximum sequence length is 20. The preprocessed statistics are presented in Table 1.

5.1.2 Baselines. Following [6, 46, 49], we compare SimRec with two types of matching baselines: general models and multi-interest models. CF-based and graph-based matching methods are excluded due to the need to model unseen users in the evaluation protocol.

- **PopRec** is a traditional method that recommends users with the most popular items.
- **YouTube DNN** [10] adopts a two-tower structure and employs average pooling to extract user representation.
- **GRU4Rec** [17] is an RNN-based model applying GRU to capture complex temporal patterns in user sequences.
- **MIND** [25] is a pioneer multi-interest work applying the dynamic routing mechanism with the capsule network.
- **ComiRec-SA** [6] employs a multi-head self-attention mechanism for multiple interest extracting. We choose the SA version, which is more robust than the DR version.
- **PIMIRec** [8] is a multi-interest model that utilizes the periodicity and interactivity in the sequence.
- **Re4** [49] is the first work introducing the backward flow into multi-interest learning.
- **REMI** [46] is an advanced multi-interest model that improves negative sample sampling and routing regularization.

5.1.3 Evaluation Protocal & Metrics. To ensure fairness, we follow the evaluation protocol of prior works [6, 46]. Based on the user count, the user sequences in datasets are partitioned into training, validation, and test sets with an 8:1:1 ratio. The entire user sequences in the training set are used for training, while in validation and testing, the first 80% of user sequences are used to model user representations, and the remaining 20% serve as target items. To evaluate the performance of our proposed methods, we employ commonly used metrics, including **Recall**, **Hit Rate**, and **NDCG**[4] (Normalized Discounted Cumulative Gain), calculated based on the top 20/50 matched candidates, following [6, 8, 46, 49].

[1] https://nijianmo.github.io/amazon
[2] https://www.kaggle.com/datasets/retailrocket/ecommerce-dataset
[3] https://www.yelp.com/dataset

[4] NDCG is computed according to ComiRec official revised code (https://github.com/THUDM/ComiRec), which adheres to the paper's NDCG calculation description but differs from the results in the original paper.

Table 3: Compatibility results. Performance of advanced multi-interest models and their SimEmb-enhanced version.

Datasets	Metric	PIMIRec	+SimEmb	Re4	+SimEmb	REMI	+SimEmb
Beauty	R@20	0.0731	**0.1082**	0.0741	**0.1010**	0.0779	**0.1190**
	R@50	0.1101	**0.1745**	0.1142	**0.1708**	0.1259	**0.1859**
	ND@20	0.0477	**0.0689**	0.0539	**0.0704**	0.0537	**0.0846**
	ND@50	0.0631	**0.0867**	0.0665	**0.0897**	0.0687	**0.1027**
	HR@20	0.1305	**0.1761**	0.1229	**0.1663**	0.1346	**0.1949**
	HR@50	0.2070	**0.2655**	0.1873	**0.2646**	0.2088	**0.2897**
Books	R@20	0.0723	**0.1031**	0.0602	**0.0728**	0.0768	**0.1013**
	R@50	0.1107	**0.1558**	0.0994	**0.1217**	0.1123	**0.1547**
	ND@20	0.0510	**0.0809**	0.0552	**0.0608**	0.0636	**0.0854**
	ND@50	0.0652	**0.0986**	0.0700	**0.0780**	0.0765	**0.1031**
	HR@20	0.1449	**0.1986**	0.1305	**0.1494**	0.1532	**0.2022**
	HR@50	0.2144	**0.2877**	0.2067	**0.0.2377**	0.2189	**0.2926**
Retail Rocket	R@20	0.4179	**0.4822**	0.4631	**0.5306**	0.4537	**0.4900**
	R@50	0.5110	**0.5923**	0.5385	**0.6310**	0.5483	**0.5979**
	ND@20	0.3643	**0.3860**	0.3580	**0.3947**	0.3877	**0.3910**
	ND@50	0.3757	**0.3989**	0.3677	**0.4072**	0.3990	**0.4036**
	HR@20	0.5912	**0.6533**	0.6226	**0.6824**	0.6283	**0.6635**
	HR@50	0.6807	**0.7515**	0.6934	**0.7714**	0.7160	**0.7569**
Yelp	R@20	0.0842	**0.1092**	0.0598	**0.1055**	0.0725	**0.1114**
	R@50	0.1487	**0.2005**	0.1183	**0.1942**	0.1328	**0.2169**
	ND@20	0.0644	**0.0876**	0.0489	**0.0787**	0.0638	**0.0871**
	ND@50	0.0861	**0.1159**	0.0695	**0.1069**	0.0852	**0.1198**
	HR@20	0.1668	**0.2130**	0.1210	**0.1984**	0.1616	**0.2229**
	HR@50	0.2784	**0.3621**	0.2270	**0.3429**	0.2732	**0.3896**

Table 4: Comparison of annotated and simulated attributes.

Datasets	Metric	ComiRec	+MaaEmb	Improv.	+SimEmb	Improv.
Beauty	R@20	0.0650	0.0741	+14.00%	**0.1139**	+75.23%
	R@50	0.1102	0.1309	+18.78%	**0.1810**	+64.25%
	ND@20	0.0415	0.0448	+7.95%	**0.0758**	+82.65%
	ND@50	0.0554	0.0612	+10.47%	**0.0927**	+67.33%
	HR@20	0.1100	0.1265	+15.00%	**0.1873**	+70.23%
	HR@50	0.1775	0.2056	+15.83%	**0.2758**	+55.38%
Books	R@20	0.0525	0.0584	+11.24%	**0.0937**	+78.48%
	R@50	0.0819	0.0913	+11.48%	**0.1408**	+71.92%
	ND@20	0.0366	0.0412	+12.57%	**0.0715**	+95.36%
	ND@50	0.0484	0.0542	+11.98%	**0.0877**	+81.20%
	HR@20	0.1088	0.1201	+10.39%	**0.1799**	+65.35%
	HR@50	0.1670	0.1844	+10.42%	**0.2607**	+56.11%
Retail Rocket	R@20	0.4133	0.4308	+4.23%	**0.4859**	+17.57%
	R@50	0.4959	0.5408	+9.08%	**0.6012**	+21.23%
	ND@20	0.3709	0.3630	-2.13%	**0.3812**	+2.78%
	ND@50	0.3829	0.3767	-1.62%	**0.3946**	+3.06%
	HR@20	0.5906	0.6057	+2.56%	**0.6600**	+11.75%
	HR@50	0.6772	0.7079	+4.53%	**0.7577**	+11.89%
Yelp	R@20	0.0625	0.0678	+8.48%	**0.0921**	+47.36%
	R@50	0.1282	0.1410	+9.98%	**0.1881**	+46.72%
	ND@20	0.0547	0.0508	-7.13%	**0.0620**	+13.35%
	ND@50	0.0787	0.0779	-1.02%	**0.0984**	+25.03%
	HR@20	0.1340	0.1361	+1.57%	**0.1839**	+37.24%
	HR@50	0.2556	0.2732	+6.89%	**0.3345**	+30.87%

5.1.4 Implementation Details. We implement our work with PyTorch 2.0 in Python 3.10. We build SimEmb on ComiRec-SA by default and name it **SimRec**, except for section 5.3. The batch size is 128 for Books dataset and 256 for other datasets. The embedding dimension is 64, and the number of interests is 4. We use Adam [22] for optimization with a learning rate 0.001. The maximum training iterations are set to 1 million rounds. The step interval threshold T for Yelp is 5, while other datasets are set at 3. We test baseline models using the open-source code provided in the original paper and reproduce REMI with PyTorch 2.0. We followed the original paper's settings for other hyperparameters of baselines. For a fair comparison, we set the number of negative samples for all models to 10*batch size.

5.2 Overall Performance

In this section, we aim to answer the question: How does SimRec perform compared to state-of-the-art matching models?

Table 2 displays the overall performance comparison on four datasets , and we have the following observations :

- General sequential recommendation models (Y-DNN and GRU4Rec) surpass PopRec, indicating that modeling the user interest based on the interaction sequence meets personalized recommendation requirements.
- Classic multi-interest models (MIND and ComiRec) outperform Y-DNN and GRU4Rec which only extract a single user representation vector, indicating the efficacy of multi-interest frameworks in representing users' various interests reflected by the item diversity in the user sequence.
- Advanced multi-interest models (PIMIRec, Re4, and REMI) achieve better performance than the classic models, with REMI standing out. Despite the primary focus of REMI not being item embedding enhancement, embeddings of irrelevant items are alienated as it revises the negative sampling, demonstrating the significance of a robust embedding space.
- SimRec, which applies our proposed SimEmb on the basic ComiRec, achieves the best performance across nearly all metrics on four datasets. Particularly, in the pivotal Recall@50 metric for the matching stage, SimRec improves baseline performances of Beauty, Books, RetailRocket, and Yelp by 46.21%, 25.38%, 9.65%, and 41.64%, respectively. While we do not lead in a few metrics, the difference from the best performance is negligible. The results reinforce the merits of item embedding enhancement with simulated attributes.

5.3 Compatibility Study

Since compatibility is one of the key factors limiting the applicability of a method, in this section, we aim to answer the question: Apart from ComiRec, how does SimEmb perform when implemented on other multi-interest models?

We compare the performance of advanced multi-interest models, PIMIRec [8], Re4 [49], and REMI [46], and their SimEmb-enhanced version. To ensure fairness, we maintain the other modules and hyperparameters in the original models unchanged and only replace the item embedding layer with the SimEmb. The comparison between the original and enhanced models is presented in Table 3.

It is evident that SimEmb notably improves the performance of all models by an average of 35.85% on Recall@20 and 38.42% on Recall@50, demonstrating the strong compatibility of SimEmb. Moreover, the substantial performance improvement across all multi-interest models further indicates their inadequate item embedding space and affirms the effectiveness of our proposed item embedding enhancement solution.

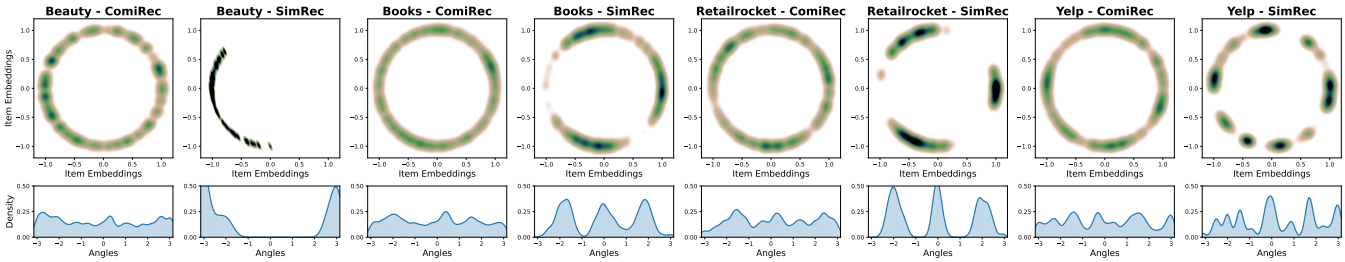

Figure 4: Item embedding distributions of ComiRec and its SimEmb-enhanced version on four datasets. Darker colors indicate denser point clustering in this area and density curve displays point densities across angles.

Table 5: Training time per batch and testing time (s).

Models	Beauty		Retail		Yelp	
	Train	Test	Train	Test	Train	Test
ComiRec	0.0071	1.1195	0.0091	1.4213	0.0082	1.0509
PIMIRec	0.0567	2.6135	0.0726	3.3171	0.0664	2.4637
Re4	0.0149	1.3676	0.0209	1.7926	0.0139	1.3463
REMI	0.0106	1.4773	0.0153	2.0594	0.0127	1.3232
SimRec	0.0108	1.1878	0.0179	1.4771	0.0134	1.0851

5.4 Performance Comparison with Manually Annotated Attribute

In this section, we aim to answer the question: How does SimEmb perform compared to utilizing manually annotated attributes?

We treat the manually annotated categories in the meta dataset as item attributes. By preserving all categories without filtering, we collect 249, 2760, 3718, and 1925 attributes from Beauty, Books, RetailRocket, and Yelp datasets, respectively. To utilize these manually annotated attributes, we embed them into embedding vectors. Then we sum the item-ID embedding with embeddings of the item's annotated attributes to obtain the enhanced item embedding. We refer to this comparison experiment as MaaEmb.

The results in Table 4 demonstrate that manually annotated attributes effectively improve the matching performance by enhancing item embeddings. However, SimEmb still surpasses, which is reasonable as the attribute information in the dataset is coarse-grained and incomplete. For example, in the Books dataset, 47.47% of items (equivalent to 286565 items) are only labeled with 'Books', which essentially equates to the absence of any category information, leading to a limited performance improvement. In contrast, SimEmb bypass manual annotation and directly simulates complete and fine-grained attribute information from interaction data, resulting in more pronounced improvement.

5.5 Hyperparameter Study

In this section, we investigate the influence of the only hyperparameter in SimEmb, the step interval threshold T. We set T from 1 to 5 to construct different co-occurrence matrices and perform experiment comparisons.

As shown in Figure 5, the Yelp dataset achieves its highest performance at a step interval threshold of 5, while the other datasets perform best with a threshold of 3. Additionally, performance diminishes when the threshold is too small or large, attributed to insufficient statistics at lower thresholds and increased noise at higher thresholds. In both cases, the co-occurrence matrices struggle to capture similarity between items effectively. Furthermore, slight fluctuations in model performance are observed when the step interval threshold changes in a proper range, emphasizing the robustness of SimEmb.

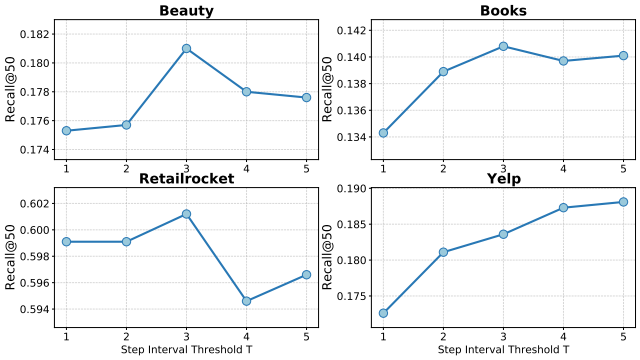

Figure 5: Performance comparison with different T.

5.6 Training and Testing Time Comparison

In this section, we report the training time on a batch and the test time on the entire test set of different multi-interest matching models. The results are collected on Intel(R) Xeon(R) Gold 5115 CPU and Quadro RTX 6000 GPU. The training time is obtained by averaging the total training time of the first 500 batches. For all models, we set the batch size as 256.

The results presented in Table 5 indicate that ComiRec has the shortest training and testing times due to its lightweight architecture. While other models that incorporate improvements within the ComiRec framework all exhibit corresponding increases in training and testing times. Among them, PIMIRec takes the longest time consumption due to its addition of a complex graph structure. On the contrary, SimRec introduces a lower complexity during the training phase, resulting in an acceptable increase in training time given its superior performance. Moreover, since we can preserve the enhanced item embedding matrix of SimeEmb after training, there is no additional time complexity introduced during testing. As a result, the testing time of SimRec is nearly on par with ComiRec.

5.7 Visualization Case Study

In this section, we present visualizations of the item embedding space in ComiRec and SimRec across four datasets, aiming to showcase the effectiveness of SimEmb intuitively.

For each dataset, we randomly select 600 items from three categories. Following the visualization method in [42], we first map the learned item embedding vectors to 2-dimensional normalized vectors on the unit hypersphere with t-SNE [40]. Then, we employ nonparametric Gaussian kernel density estimation (KDE) [4] to plot the embedding distributions in $\mathbb{R}^2$ and von Mises-Fisher (vMF) KDE to plot the density curve of embedding distributions on angles.

As shown in Figure 4, the item embeddings of ComiRec exhibit uniform distributions across the circle, with relatively flat density curves. It suggests the clustering deficiency within the item embedding space, which limits the model's matching performance. In contrast, the item embeddings of SimRec form clusters in narrow arcs accompanied with sharper density curves, showcasing a notable improvement of clustering in the item embedding space. This improvement can be attributed to the weighted summation of simulated attributes, where items with shared attributes tend to have similar embeddings, automatically promoting clustering.

6 RELATED WORK

6.1 Deep Candidate Matching

With the rapid advancement of deep learning, lightweight neural networks have also gained widespread utilization in the matching stage. Initial methods grounded in collaborative filtering (CF) [21, 29, 36]. Neural CF [16] harnessed matrix factorization components and a non-linear multi-layer perceptron (MLP) to effectively model the matching between users and items. As the field progressed, graph-based recommendation [2, 12, 14, 23, 26, 33, 34, 39] and sequential recommendation [18, 38, 47, 48] are introduced into the matching stage. The graph-based model amasses insights from neighbor nodes within the user-item graph to refine the node representation of users and items. LightGCN [15] and SGC [43] streamline the GCN structure and redundant operations, showcasing an efficient and effective approach that inspired subsequent graph-based recommendation models like SGL [44]. The sequential recommendation explore the dynamic user interest information inherent within the historical interaction sequence. YouTube-DNN [10] is an early pioneer in sequential recommendation, employing a dual-tower architecture and deriving user characteristics via average pooling of interacted items. Drawing from natural language processing (NLP), subsequent works adopt frameworks such as recurrent neural networks [17] and memory networks [9].

6.2 Multi-interest Recommendation

Recent studies in sequential recommendation have found that extracting a unified user representation vector from sequences is insufficient to capture the multiple interests of the user. As this concern is more prominent in the matching stage, multi-interest recommendations have gained increasing attention. MIND [25] pioneers using a dynamic routing mechanism to extract multiple interest vectors from the user sequence. ComiRec [6] introduces a framework based on the multi-head attention mechanism and harmonizes the trade-off between recommendation diversity and precision. PIMIRec [8] considers the periodicity and interactivity within sequences. UMI [7] and UDM [13], respectively, leverage user profiles and target items to assist in extracting multiple interest vectors. Re4 [49] introduces the backward flow to reevaluate interest embeddings. REMI [46] questions the uniformly sampled softmax and routing collapse in the multi-interest framework, proposing an improved negative sampling strategy and the routing regularization method. Despite these works making improvements in the multi-interest extraction module and loss functions from various perspectives, few efforts are devoted to enhancing item embeddings, which limits the effectiveness of item matching.

6.3 Attribute Information Completion in Recommendation

In practical scenarios, the attribute side information of items may be incomplete or unavailable, posing challenges for attribute fusion in the recommendation. Traditional methods for side information completion fill the missing attribute with heuristic values like average or randomized values [3, 24, 37]. However, these context-independent values can perturb the training of model parameters. Another approach is to estimate the values of the missing attribute. Early models based on auto-encoders (AE) [1, 32, 41] apply random dropout to the side information and reconstruct the lost data. KTUP [5] predicts missing relationships within the knowledge graph to introduce knowledge into the recommendation. CC-CC [37] calculates the importance of features and proposes an adaptive feature sampling strategy to enhance model robustness when dealing with missing features, avoiding direct prediction of missing data. AGCN [45] cyclically updates the graph using approximated attribute values to boost attribute inference and item recommendation. MIIR [28] unifies the tasks of missing attribute information completion and sequential recommendation.

In contrast to our work, these methods have two limitations: 1) The lack of specific design for the matching stage leading to the issue of high complexity; 2) The attribute information completion depends on manually annotated attributes and cannot handle cases where attribute information is unavailable.

7 CONCLUSION

In this work, we pinpoint the inadequate item embedding of prior multi-interest models in the matching stage. Building upon the theoretical feasibility, we propose a simple yet effective module, SimEmb, which employs co-occurrence matrices to simulate attribute information for item embedding enhancement. Extensive experiments and visualization demonstrate the superiority of SimEmb in both accuracy and complexity. In the future, we will tackle the limitations of our work by focusing on denoising methods during the construction of the co-occurrence matrix, as the unavoidable introduction of noise interferes with the approximation of item similarity.

We believe our work offers a simple and reliable alternative for the labor-intensive task of manually annotating attributes with broader implications beyond multi-interest recommendation, and can extend to other recommendation fields, thereby inspiring future research.

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China (NSFC) (61972455) and the Project of Science and Technology Research and Development Plan of China Railway Corporation (N2023J044).

ETHICAL CONSIDERATIONS

We acknowledge the importance of ethical considerations and potential societal impacts, and our method doesn't exhibit apparent negative consequences. We assure stringent data anonymization and compliance with privacy regulations. Our method's design emphasizes responsible deployment and protect user privacy in alignment with ethical principles.

REFERENCES

[1] Brett K Beaulieu-Jones, Jason H Moore, and POOLED RESOURCE OPEN-ACCESS ALS CLINICAL TRIALS CONSORTIUM. 2017. Missing data imputation in the electronic health record using deeply learned autoencoders. In *Pacific symposium on biocomputing 2017*. World Scientific, 207–218.

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

[3] Felix Bießmann, David Salinas, Sebastian Schelter, Philipp Schmidt, and Dustin Lange. 2018. "Deep" Learning for Missing Value Imputationin Tables with Non-Numerical Data. In *CIKM*. 2017–2025.

[4] Zdravko I. Botev, Joseph F. Grotowski, and Dirk P. Kroese. 2010. Kernel density estimation via diffusion. *Annals of Statistics* 38 (2010), 2916–2957.

[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*. 151–161.

[6] Yukuo Cen, Jianwei Zhang, Xu Zou, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Controllable multi-interest framework for recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2942–2951.

[7] Zheng Chai, Zhihong Chen, Chenliang Li, Rong Xiao, Houyi Li, Jiawei Wu, Jingxu Chen, and Haihong Tang. 2022. User-aware multi-interest learning for candidate matching in recommenders. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1326–1335.

[8] Gaode Chen, Xinghua Zhang, Yanyan Zhao, Cong Xue, and Ji Xiang. 2021. Exploring Periodicity and Interactivity in Multi-Interest Framework for Sequential Recommendation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*. 1426–1433.

[9] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 108–116.

[10] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.

[11] Carlos A Gomez-Uribe and Neil Hunt. 2015. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)* 6, 4 (2015), 1–19.

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[13] Long Guo, Fei Fang, Binqiang Zhao, and Bin Cui. 2022. UDM: A Unified Deep Matching Framework in Recommender Systems. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 3122–3130.

[14] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[15] 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.

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

[17] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

[18] Jui-Ting Huang, Ashish Sharma, Shuying Sun, Li Xia, David Zhang, Philip Pronin, Janani Padmanabhan, Giuseppe Ottaviano, and Linjun Yang. 2020. Embedding-based retrieval in facebook search. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2553–2561.

[19] Sébastien Jean, KyungHyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On Using Very Large Target Vocabulary for Neural Machine Translation. In *ACL*. 1–10.

[20] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data* 7, 3 (2019), 535–547.

[21] Santosh Kabbur, Xia Ning, and George Karypis. 2013. FISM: factored item similarity models for top-N recommender systems. In *KDD*. ACM, 659–667.

[22] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[23] Thomas Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *ArXiv* abs/1609.02907 (2016). https://api.semanticscholar.org/CorpusID:3144218

[24] Youngnam Lee, Sang-Wook Kim, Sunju Park, and Xing Xie. 2018. How to impute missing ratings? Claims, solution, and its application to collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference*. 783–792.

[25] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-interest network with dynamic routing for recommendation at Tmall. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 2615–2623.

[26] Houyi Li, Zhihong Chen, Chenliang Li, Rong Xiao, Hongbo Deng, Peng Zhang, Yongchao Liu, and Haihong Tang. 2021. Path-based deep network for candidate item matching in recommenders. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1493–1502.

[27] Yile Liang, Tieyun Qian, Qing Li, and Hongzhi Yin. 2021. Enhancing Domain-Level and User-Level Adaptivity in Diversified Recommendation. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2021).

[28] Yujie Lin, Zhumin Chen, Zhaochun Ren, Chenyang Wang, Qiang Yan, Maarten de Rijke, Xiuzhen Cheng, and Pengjie Ren. 2023. Modeling Sequential Recommendation as Missing Information Imputation. *arXiv preprint arXiv:2301.01762* (2023).

[29] Greg Linden, Brent Smith, and Jeremy York. 2003. Industry report: Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Distributed Systems Online* 4, 1 (2003), 76–80.

[30] Fuyu Lv, Taiwei Jin, Changlong Yu, Fei Sun, Quan Lin, Keping Yang, and Wilfred Ng. 2019. SDM: Sequential deep matching model for online large-scale recommender system. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2635–2643.

[31] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

[32] Ricardo Cardoso Pereira, Miriam Seoane Santos, Pedro Pereira Rodrigues, and Pedro Henriques Abreu. 2020. Reviewing autoencoders for missing data imputation: Technical trends, applications and outcomes. *Journal of Artificial Intelligence Research* 69 (2020), 1255–1285.

[33] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 701–710.

[34] Tieyun Qian, Yile Liang, Qing Li, and Hui Xiong. 2022. Attribute Graph Neural Networks for Strict Cold Start Recommendation. *IEEE Transactions on Knowledge and Data Engineering* 34 (2022), 3597–3610.

[35] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. *Advances in neural information processing systems* 30 (2017).

[36] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[37] Shaoyun Shi, Min Zhang, Xinxing Yu, Yongfeng Zhang, Bin Hao, Yiqun Liu, and Shaoping Ma. 2019. Adaptive feature sampling for recommendation with missing content feature values. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1451–1460.

[38] K. Sun, Tieyun Qian, Hongzhi Yin, Tong Chen, Yiqi Chen, and Ling Chen. 2019. What Can History Tell Us? *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (2019).

[39] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*. 1067–1077.

[40] Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605.

[41] Cheng Wang, Mathias Niepert, and Hui Li. 2018. LRMM: Learning to Recommend with Missing Modalities. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, 3360–3370.

[42] Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*. 9929–9939.

[43] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International*

conference on machine learning. 6861–6871.

[44] 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.

[45] Le Wu, Yonghui Yang, Kun Zhang, Richang Hong, Yanjie Fu, and Meng Wang. 2020. Joint item recommendation and attribute inference: An adaptive graph convolutional network approach. In *Proceedings of the 43rd International ACM SIGIR conference on Research and Development in Information Retrieval*. 679–688.

[46] Yueqi Xie, Jingqi Gao, Peilin Zhou, Qichen Ye, Yining Hua, Jaeboum Kim, Fangzhao Wu, and Sunghun Kim. 2023. Rethinking Multi-Interest Learning for Candidate Matching in Recommender Systems. *arXiv preprint arXiv:2302.14532* (2023).

[47] Ji Yang, Xinyang Yi, Derek Zhiyuan Cheng, Lichan Hong, Yang Li, Simon Xiaoming Wang, Taibai Xu, and Ed H Chi. 2020. Mixed negative sampling for learning two-tower neural networks in recommendations. In *Companion Proceedings of the Web Conference 2020*. 441–447.

[48] Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Kumthekar, Zhe Zhao, Li Wei, and Ed Chi. 2019. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 269–277.

[49] Shengyu Zhang, Lingxiao Yang, Dong Yao, Yujie Lu, Fuli Feng, Zhou Zhao, Tatseng Chua, and Fei Wu. 2022. Re4: Learning to re-contrast, re-attend, re-construct for multi-interest recommendation. In *Proceedings of the ACM Web Conference 2022*. 2216–2226.