

Article

Multi-Channel Hypergraph Collaborative Filtering with Attribute Inference

Yutong Jiang ¹, Yuhan Gao ^{2,3}, Yaoqi Sun ^{2,3}, Shuai Wang ^{2,3} and Chenggang Yan ^{2,*}

¹ School of Mechanical, Electrical, and Information Engineering, Shandong University, Weihai 264209, China; jiangyutong@mail.sdu.edu.cn

² Department of Automation, Hangzhou Dianzi University, Hangzhou 310018, China; yuhangao@hdu.edu.cn (Y.G.); syq@hdu.edu.cn (Y.S.); shuaiwang.tai@gmail.com (S.W.)

³ Lishui Institute of Hangzhou Dianzi University, Lishui 323000, China

* Correspondence: cgyan@hdu.edu.cn

Abstract: In the field of collaborative filtering, attribute information is often integrated to improve recommendations. However, challenges remain unaddressed. Firstly, existing data modeling methods often fall short of appropriately handling attribute information. Secondly, attribute data are often sparse and can potentially impact recommendation performance due to the challenge of incomplete correspondence between the attribute information and the recommendations. To tackle these challenges, we propose a hypergraph collaborative filtering with attribute inference (HCFA) framework, which segregates attribute and user behavior information into distinct channels and leverages hypergraphs to capture high-order correlations among vertices, offering a more natural approach to modeling. Furthermore, we introduce behavior-based attribute confidence (BAC) for assessing the reliability of inferred attributes concerning the corresponding behaviors and update the most credible portions to enhance recommendation quality. Extensive experiments conducted on three public benchmarks demonstrate the superiority of our model. It consistently outperforms other state-of-the-art approaches, with ablation experiments further confirming the effectiveness of our proposed method.

Keywords: hypergraph learning; collaborative filtering; attribute inference



Citation: Jiang, Y.; Gao, Y.; Sun, Y.; Wang, S.; Yan, C. Multi-Channel Hypergraph Collaborative Filtering with Attribute Inference. *Electronics* **2024**, *13*, 903. <https://doi.org/10.3390/electronics13050903>

Academic Editor: Ping-Feng Pai

Received: 29 January 2024

Revised: 17 February 2024

Accepted: 24 February 2024

Published: 27 February 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In collaborative filtering (CF), leveraging attribute information can significantly enhance the recommendation process by capturing the similarities among users and items [1–4]. For instance, students might prefer stationery to people with other occupations, and children may favor cartoons over horror movies. These attributes provide a multi-faceted description that can reflect user preferences and behavior, thus enriching personalized recommendations [5–7].

Two pressing challenges in the realm of attribute information are its proper modeling and the challenge of sparsity [8,9]. Addressing the first challenge involves considering how attributes are integrated into collaborative filtering models. One common practice is to incorporate attributes directly into user and item embeddings [10–12]. However, this direct incorporation can be problematic if individual behaviors frequently diverge from stereotypical patterns linked to these attributes. An overemphasis on stereotypical attribute–behavior alignments may lead to inaccurate recommendations, as it fails to capture the full complexity and variability of human behavior. Another approach involves using graph-based models to represent attribute information, as employed by methods in [10,11,13]. While these models are effective in capturing pairwise relationships, they may not fully represent the group traits often characterized by attribute information, thus not fully exploiting the high-order correlations that attributes can provide. To address the second challenge of attribute sparsity, methods like those in [10] use attribute inference to

generate pseudo-labels for adaptive updating. However, this strategy risks propagating errors through incorrect pseudo-labels, leading to increasingly distorted representations as learning progresses.

To solve these issues, we propose a **hypergraph collaborative filtering with attribute inference (HCFA)** framework. HCFA effectively models attribute information, resolves the issue of attribute sparsity, and prevents error propagation. First, HCFA employs a multi-channel approach to separately model attribute and behavior information using a channel attention mechanism to assign distinct weights to each channel. This strategy allows users or items to select attributes that more accurately reflect their behaviors, thereby avoiding the pitfalls of incorporating irrelevant or inaccurate attributes into their embeddings. Second, we leverage hypergraph structures to capture the high-order correlations between attributes and behaviors. The hypergraph's ability to encompass any number of vertices makes it ideal for representing the complex, group-based relationships inherent in attribute and behavior data. Finally, to tackle attribute sparsity with attribute inference and mitigate error propagation, we propose the behavior-based attribute confidence (BAC). BAC quantitatively assesses an attribute's credibility in relation to the corresponding user or item behavior, calculated as the average similarity of the attribute with the neighbors of the user or item. Attributes with higher BAC scores are prioritized for updates in subsequent training rounds.

To conclude, our contributions are summarized as follows:

- We propose the **hypergraph collaborative filtering with attribute inference (HCFA)** framework, leveraging a multi-channel hypergraph to distinctly model attribute and behavior information. This approach enables the extraction of high-order correlations, providing precise modeling for both dimensions.
- We introduce the **behavior-based attribute confidence (BAC)** metric to quantitatively ascertain the credibility of inferred attributes. Coupled with a selection and update mechanism, this approach prioritizes attributes with higher BAC scores, thus resolving the sparsity issue and circumventing error propagation.
- Our experimental results on three public benchmarks show that our HCFA framework achieves an average performance improvement of 9.23% compared with state-of-the-art methods. Extensive ablation studies further demonstrate the effectiveness of each component of our method.

2. Related Work

The following related works are organized in Table 1.

Table 1. Related methods. The check mark (✓) indicates that the method utilizes the corresponding structure or information.

Methods	Graph	Hypergraph	Attribute
NGCF	✓	-	-
GC-MC	✓	-	-
PinSage	✓	-	-
SGCN	✓	-	-
LightGCN	✓	-	-
DHCF	-	✓	-
HCCF	-	✓	-
AGCN	✓	-	✓
AGNN	✓	-	✓
BSAL	-	-	✓
BiANE	-	-	✓

2.1. Graph Collaborative Filtering Methods

Graph convolutional networks (GCNs) [14,15] have garnered significant successes in vertex representation learning, leading to their integration into CF frameworks such as NGCF [16], GC-MC [17], and PinSage [18]. To streamline these models for CF, methods like SGCN [19] and LightGCN [20] have simplified the GCN architecture. Notably, LightGCN simplifies the GCN approach by solely focusing on neighbor aggregation, thus enhancing performance, reducing complexity, and establishing itself as state-of-the-art. However, while graph structures adeptly capture pairwise relationships, they often fall short of encapsulating high-order correlations, such as those inherent in attribute information.

2.2. Hypergraph Collaborative Filtering Methods

Compared to traditional graph structures that primarily model pairwise correlations, the hypergraph structure is notably proficient at capturing the intricate, high-order relationships among users and items. As deep learning continues to evolve, a number of methodologies utilizing hypergraphs for collaborative filtering have emerged [21–25]. Among them, DHCF [21] stands out as the first CF method based on hypergraph convolutional networks, utilizing hypergraph convolution across dual channels to refine user and item embeddings independently. Building on this, HCCF [22] advances the field with a hypergraph-enhanced cross-view contrastive learning architecture adept at integrating local and global collaborative relationships. This model synergizes hypergraph structure learning with self-supervised learning, significantly improving the discriminative capabilities of GNN-based CF models and, consequently, the representational quality within recommender systems.

2.3. Recommendation with Attribute Information

In the realm of CF, augmenting models with user and item attributes has proven to be an effective strategy for overcoming data sparsity. AGCN [10] leverages a GCN-based adaptive model for attribute inference and updates attributes to address the sparsity issue. On a different front, AGNN [13] employs a variational auto-encoder framework to generate preference embeddings that are particularly useful for cold-start users or items. Meanwhile, BSAL [26] capitalizes on node attributes to construct a semantic topology, extracting structural insights from the resulting semantic graph. Similarly, BiANE [12] focuses on integrating attribute proximity with structural information, thus enhancing node representation for more precise recommendations. While these methods innovatively incorporate attribute information into collaborative filtering, their approach to modeling attributes is not without drawbacks, potentially leading to degradation. Moreover, AGCN's [10] strategy of updating attribute information to combat sparsity issues, despite being creative, introduces the risk of error propagation, which could compound inaccuracies over time.

3. Preliminaries

A hypergraph is the generalization of a graph, where edges can connect any number of vertices [27]. Formally, a hypergraph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where vertex set $\mathcal{V}$ and hyperedge set $\mathcal{E}$ are finite sets. Each element $e \in \mathcal{E}$ is a non-empty subset of $\mathcal{V}$. The incidence matrix of a hypergraph is defined as $\mathbf{H} \in \{0, 1\}^{|\mathcal{V}| \times |\mathcal{E}|}$. It is used to represent interactions between the vertex set $\mathcal{V}$ and the hyperedge set $\mathcal{E}$. It can be formulated as

$$\mathbf{H}(v, e) = \begin{cases} 1, & v \in e \\ 0, & v \notin e \end{cases} \quad (1)$$

The degree of each vertex v in a hypergraph $\mathcal{G}$ can be defined as $d(v) = \sum_{e \in \mathcal{E}} \mathbf{H}(v, e)$, and the degree of each hyperedge e can be defined as $\delta(e) = \sum_{v \in \mathcal{V}} \mathbf{H}(v, e)$. Furthermore, $\mathbf{D}_v \in \mathbb{N}^{|\mathcal{V}| \times |\mathcal{V}|}$ and $\mathbf{D}_e \in \mathbb{N}^{|\mathcal{E}| \times |\mathcal{E}|}$ represent the diagonal matrix of the vertex and hyperedge degrees, respectively.

The hyperedge convolutional layer [28] is defined as

$$\mathbf{X}^{(l+1)} = \sigma(\mathbf{D}_v^{-1/2} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^T \mathbf{D}_v^{-1/2} \mathbf{X}^{(l)} \Theta^{(l)}), \quad (2)$$

where $\mathbf{X}^{(l)} \in \mathbb{R}^{n \times d}$ is the signal of the hypergraph at l layer, and n and d denote the vertex number and embedding dimension, respectively. σ denotes the nonlinear activation function. $\mathbf{X}^{(l+1)}$ is the output of layer $l + 1$, which can be used for a variety of downstream tasks.

In summary, compared to a graph, a hypergraph proves more effective by enabling advanced information interaction through its structural superiority.

4. Method

In this section, we introduce the proposed hypergraph collaborative filtering with attribute inference (HCFA) framework, as illustrated in Figure 1.

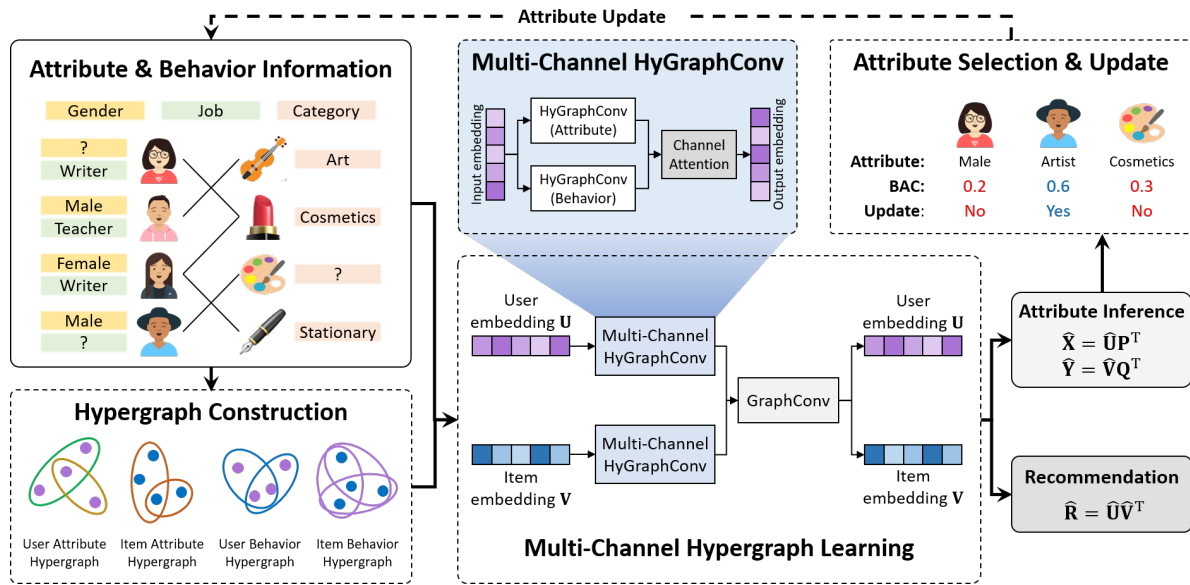


Figure 1. The pipeline of the proposed method. The initial attribute and behavior information, along with the constructed hypergraph structure, feed into the multi-channel hypergraph learning module. The learned embeddings are leveraged for attribute information and recommendation, where the former is further sent for attribute selection and update and the latter is the output of the proposed method.

4.1. Hypergraph Construction

To capture the high-order correlation between users and items, we utilize both attribute and behavioral information to construct hypergraphs.

First, the attribute hypergraph is built to capture the high-order relationship established by the attributes shared among users and items. Vertices sharing the same attribute are linked by a hyperedge. This is noted as A , and A_u , A_v represent the attribute hypergraphs for the user and item, respectively. The incidence matrix can be formulated as

$$\begin{aligned} H_A^u(i, j) &= \begin{cases} 1 & \text{user}_i \text{ have attribute}_j \\ 0 & \text{otherwise} \end{cases}, \\ H_A^v(i, j) &= \begin{cases} 1 & \text{item}_i \text{ have attribute}_j \\ 0 & \text{otherwise} \end{cases}. \end{aligned} \quad (3)$$

Thus, $H_A^u = X$ and $H_A^v = Y$.

Next, the behavior hypergraph is built to capture the high-order relationship established by the behavior information. Specifically, users who interact with the same item will be grouped into the same hyperedge. Similarly, items that interact with the same user will be linked by the same hyperedge. This is noted as B for the sake of brevity, and B_u, B_v represent the behavior hypergraphs for the user and item, respectively. The incidence matrix can be formulated as

$$\begin{aligned} H_B^u(i, j) &= \begin{cases} 1 & \text{user}_i \text{ interacted with item}_j \\ 0 & \text{otherwise} \end{cases}, \\ H_B^v(i, j) &= \begin{cases} 1 & \text{item}_i \text{ interacted with user}_j \\ 0 & \text{otherwise} \end{cases}. \end{aligned} \quad (4)$$

Obviously, it can be found out that $H_B^u = R$ and $H_B^v = R^T$, where R is the user-item interaction matrix.

4.2. Multi-Channel Hypergraph Learning

After generating the two aforementioned types of hypergraphs for users and items, multi-channel hypergraph convolution is applied using user and item embeddings.

Multi-channel hypergraph convolution. Inspired by [20], the nonlinear activation layer and the learnable linear transformation parameters W and Θ are removed as they might cause precision loss. Therefore, the hypergraph convolution is performed based on

$$X^{(k+1)} = D_v^{-1/2} H D_e^{-1} H^T D_u^{-1/2} X^{(k)}, \quad (5)$$

where $X^{(k)}$ denotes the vertex embedding after k layer convolution. In our case, it can be replaced by $U_c^{(k)}$ and $V_c^{(k)}$ for the user and item embeddings.

After propagating the user and item embeddings through K layers, the results from each layer are averaged to obtain the final result of each channel, thereby mitigating the over-smoothing problem [20]:

$$u_c^* = \frac{1}{K+1} \sum_{l=0}^K u_c^{(l)}, v_c^* = \frac{1}{K+1} \sum_{l=0}^K v_c^{(l)}. \quad (6)$$

Channel attention. Furthermore, a channel attention operation is utilized on the final generated channel-specific user and item representations to aggregate information from different channels to form comprehensive user and item representations. According to [29], the attention operation is defined as

$$\begin{aligned} \alpha_u^c &= f_a(u_c^*) = \frac{\exp(a_u^T \cdot W_a^{c,u} u_c^*)}{\sum_c \exp(a_u^T \cdot W_a^{c,u} u_c^*)}, \\ \alpha_v^c &= f_a(v_c^*) = \frac{\exp(a_v^T \cdot W_a^{c,v} v_c^*)}{\sum_c \exp(a_v^T \cdot W_a^{c,v} v_c^*)}, \end{aligned} \quad (7)$$

where $W_a \in \mathbb{R}^{d \times d}$ is a trainable parameter. Additionally, comprehensive representations of users and items after channel attention can be formulated as

$$u^* = \sum_c \alpha_u^c u_c^*, v^* = \sum_c \alpha_v^c v_c^*, \quad (8)$$

where $c \in \{A_u, B_u, A_v, B_v\}$.

Graph convolution. To further leverage implicit interactive information between users and items, we conduct a graph convolution on the representations of users and items

to extract more pairwise relationship information. Following the definitions in [14,20], the graph convolution layer is defined as

$$\begin{pmatrix} \mathbf{U}^{*(j+1)} \\ \mathbf{V}^{*(j+1)} \end{pmatrix} = \mathbf{D}^{-1/2} \begin{pmatrix} \mathbf{0} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{0} \end{pmatrix} \mathbf{D}^{-1/2} \begin{pmatrix} \mathbf{U}^{*(j)} \\ \mathbf{V}^{*(j)} \end{pmatrix}, \quad (9)$$

where $\mathbf{R} \in \mathbb{R}^{n_u \times n_v}$ is the interaction matrix between n_u users and n_v items, and $\mathbf{D}$ is a $(n_u + n_v) \times (n_u + n_v)$ diagonal degree matrix. After propagating the embedded feature through J graph convolution layers, the results obtained in each layer will be averaged to obtain the final result. This can be formulated as

$$\hat{\mathbf{U}} = \frac{1}{J+1} \sum_{l=0}^J \mathbf{U}^{*(l)}, \quad \hat{\mathbf{V}} = \frac{1}{J+1} \sum_{l=0}^J \mathbf{V}^{*(l)}. \quad (10)$$

This is the same operation as Equation (6) mentioned above.

4.3. Recommendation and Attribute Inference

First, the recommendation task can be performed according to the idea of matrix factorization [30]. The user and item interaction matrix can be predicted as

$$\hat{\mathbf{R}} = \hat{\mathbf{U}} \hat{\mathbf{V}}^T. \quad (11)$$

Next, to improve the attribute inference process, we consider both attributes and vertices as entities in our matrix factorization framework. This approach enables us to embed attributes into the same semantic space as users and items, simplifying the computation of similarities across all entities. By placing attribute embeddings in a shared space, correlations among users, items, and attributes themselves can be more precisely established. The matrix factorization is formulated as follows:

$$\hat{\mathbf{X}} = \hat{\mathbf{U}} \mathbf{P}^T, \quad \hat{\mathbf{Y}} = \hat{\mathbf{V}} \mathbf{Q}^T, \quad (12)$$

where $\hat{\mathbf{X}} \in \mathbb{R}^{n_u \times n_p}$ and $\hat{\mathbf{Y}} \in \mathbb{R}^{n_v \times n_q}$ represent the inferred attribute matrices for users and items, respectively, while $\mathbf{P} \in \mathbb{R}^{n_p \times d}$ and $\mathbf{Q} \in \mathbb{R}^{n_q \times d}$ denote the corresponding attribute embeddings. This bidirectional flow of information between the vertices and the attributes not only enriches the attribute data but also underscores the individual importance of each attribute in relation to different vertices. Consequently, the use of matrix factorization for inferring attributes is both logical and effective.

4.4. Optimization

First, the Bayesian personalized ranking (BPR) loss is utilized to optimize the recommendation outcomes. This loss function is a pairwise personalized ranking loss derived from the maximum posterior estimator and is widely employed in various recommendation models. Its formulation is as follows:

$$\mathcal{L}_r = \sum_{j \in \mathcal{I}(i), k \notin \mathcal{I}(i)} -\log \sigma(\hat{r}_{i,j} - \hat{r}_{i,k}) + \lambda \|\Phi_r\|_2^2, \quad (13)$$

where Φ_r represents the parameters of the model, $\hat{r}_{i,j} = u_i^T v_j$ is the prediction score of user i on item j , and $\sigma(\cdot)$ represents the sigmoid function.

Next, the cross-entropy (CE) loss is leveraged to optimize the attribute inference, as it can be regarded as classification. It is formulated as

$$\begin{aligned} \mathcal{L}_i^u &= -\frac{1}{n_u} \sum_i \sum_{j=1}^{n_p} x_{ij} \log(\hat{x}_{ij}), \\ \mathcal{L}_i &= \mathcal{L}_i^u + \mathcal{L}_i^v, \end{aligned} \quad (14)$$

where $\hat{x}_{i,j} = u_i^T p_j$ is the inference score of user_{*i*} on user attribute_{*j*}, and $\hat{y}_{i,j} = v_i^T q_j$ is the inference score of item_{*i*} on item attribute_{*j*}.

Therefore, the overall loss function of the proposed method can be written as

$$\mathcal{L} = \mathcal{L}_r + \gamma \cdot \mathcal{L}_i, \quad (15)$$

where γ is a hyper-parameter to balance the two different losses.

4.5. Attribute Selection and Update

To address the issue of sparse attribute information, we adopt the attribute updating strategy from AGCN [10]. However, this approach carries the risk of error propagation, where inaccurately inferred attributes might be carried over into subsequent training iterations, potentially amplifying the initial errors. Furthermore, the informativeness of attributes for recommendations varies, as some truthful attributes may not necessarily indicate behavior patterns. There can be instances of atypical consumer behavior that deviates from expected norms, such as individuals frequently engaging with products traditionally associated with a different demographic. For example, a professional who frequently purchases items categorized outside of their field of expertise or a customer who consistently prefers products that are not traditionally aligned with their demographic profile.

Therefore, to better assess the relevance of attributes in the context of user and item behaviors, we introduce the concept of **behavior-based attribute confidence (BAC)**. This metric quantifies the confidence in a vertex's attribute by considering its associated behaviors. As previously discussed, users and items, along with their respective attributes, are represented as vertices within a shared semantic space. This representation facilitates the computation of similarities between any pair of vertices. The BAC's definition and calculation are thus detailed below:

Definition 1 (Behavior-based attribute confidence (BAC)). *The confidence of an attribute with respect to a vertex reflects the degree of alignment between the attribute and the vertex's behavior, as expressed by the average similarity between the attribute and all neighboring nodes of the given vertex.*

Given a node v and an attribute a , the behavior-based attribute confidence $BAC(a, v)$ is calculated as follows:

$$BAC(a, v) = \frac{1}{|N(v)|} \sum_{u \in N(v)} s(a, u), \quad (16)$$

where $N(v)$ is the set of neighbors of v , and $s(a, u)$ is the similarity score between attribute a and neighbor u .

After calculating the BAC for each inferred attribute, we select those with high BAC values to update in the subsequent training iterations. This selection process is regulated by the hyper-parameter β , which has a range from 0 to 1. A β value of 0 means that no attributes are updated, while a value of 1 indicates that all inferred attributes are subject to an update. The frequency of these updates is determined by the parameter L , where $L = 0$ implies that no updates are performed.

5. Experiments

In our experimental evaluation, we address several research questions that cumulatively build a comprehensive understanding of our proposed method's performance and characteristics:

- Q1.** *Is the behavior-based attribute confidence (BAC), as delineated in Definition 1, an accurate and reliable measure for quantifying the extent to which a vertex's attribute is pertinent to its behaviors?*
- Q2.** *How does the proposed modeling method perform compared to existing methods?*
- Q3.** *How effective is each component of the model?*

Q4. How does parameter sensitivity affect the model's performance?

5.1. Experimental Protocol

5.1.1. Datasets, Metrics, and Settings

Datasets. We conduct experiments on three public benchmarks: MovieLens (<https://grouplens.org/datasets/movielens/1m/> accessed on 30 October 2023) [31], RentTheRunway (https://cseweb.ucsd.edu/~jmcauley/datasets.html#clothing_fit accessed on 30 October 2023) [32], and Google Local Reviews (https://cseweb.ucsd.edu/~jmcauley/datasets.html#google_local accessed on 30 October 2023) [33,34]. These datasets contain behavior and attribute information, as shown in Table 2.

Table 2. Datasets details.

Dataset	Users	Items	Ratings	Sparsity	User Attributes	Item Attributes
MovieLens	6040	3952	226,310	99.052%	Gender, age, occupation, zip code	Genre
RentTheRunway	1194	11,310	20,853	99.846%	Age, body type, height, weight	Category, size
Google Local Reviews	1656	1380	8901	99.611%	Job	Category, address

We randomly split the dataset into the training set and the testing set with a ratio of 9 to 1. For each type of attribute, we randomly delete 90% and keep only 10% for training to ensure sparsity. The deleted attributes are only used for testing.

Metrics. For evaluating the top- k recommendation performance on ranking tasks of different methods, we use four commonly used evaluation metrics, i.e., precision@ k , recall@ k , normalized discounted cumulative gain (ndcg@ k), and hit ratio (hr@ k). In our experiments, k is set to 10.

Settings. For a fair comparison, we initialize all latent embeddings using the Xavier uniform initializer, and employ Adam [35] for optimization with a learning rate of 0.001. For our model, the number of latent factors is set to 32, and the batch size is set to 1024 for all models. The number of hypergraph convolution layers K and the number of graph convolution layers J are both set to 2. The number of updates L is set to 20. For the three datasets, the hyper-parameter γ in Equation (15) is set to 0.01, 0.001, and 0.001, respectively, while β is set to 0.8, 0.4, and 0.2, respectively. All experiments were conducted using the Python programming language and the PyTorch framework on a server equipped with two Intel Xeon E5-2678 2.50 GHz CPUs and an Nvidia GeForce RTX 3090 GPU.

5.1.2. Baselines

We compare our method with the following state-of-the-art methods. They can be categorized into three groups.

Graph collaborative filtering methods:

- **NGCF** [16]: A classical model that integrates the bipartite graph structure of user–item interaction into the embedding process.
- **LightGCN** [20]: A state-of-the-art graph-based collaborative filtering method that simplifies and removes unnecessary parts of GCN.

Hypergraph collaborative filtering methods:

- **DHCF** [21]: A hypergraph-based method that proposes the jump hypergraph convolution (JHConv) method to support the explicit and efficient embedding propagation of high-order correlations.
- **HCCF** [22]: A self-supervised recommendation framework that jointly captures local and global collaborative relations with a hypergraph-enhanced cross-view contrastive learning architecture.

Methods incorporating attribute information:

- **AGCN [10]**: An adaptive graph convolutional network approach for joint item recommendation and attribute inference that could adjust the graph embedding learning parameters by incorporating both the given attributes and the estimated attribute values.
- **BiANE [12]**: A bipartite attributed network embedding method that models both the inter-partition proximity and the intra-partition proximity.

5.2. Validity of Behavior-Based Attribute Confidence (Q1)

In this section, our experiments are designed to assess the validity of the proposed behavior-based attribute confidence (BAC), as outlined in Definition 1. The procedure commences with the training of the base model using a complete dataset to establish a BAC score for each attribute. Subsequently, we conduct two sets of experiments to demonstrate the model.

Experiments with BAC quintiles. Attributes are stratified based on their BAC into five distinct categories. These categories are ordered by BAC and are segmented into quintiles along the x-axis, denoted by “ n ”. For instance, $n = 1$ represents attributes in the top 20% of BACs, while $n = 5$ corresponds to the bottom 20%. The blue bars in Figure 2 represent the NDCG values for each quintile category. This visualization allows us to observe how the recommendation performance varies with the attributes of differing BACs. The red line across the quintiles depicts the average BAC within each category, providing a clear visual correlation between the BAC and the recommendation performance. The analysis reveals a discernible trend where higher BAC categories ($n = 1, 2$), which represent attributes with stronger confidence, correlate with higher NDCG values. Conversely, as the BAC average decreases across the quintiles ($n = 3, 4, 5$), there is a noticeable decline in NDCG performance. This trend demonstrates the efficacy of the BAC as a reliable indicator of attribute quality and its consequent impact on recommendation performance. By employing attributes with higher BACs, our HCFA model effectively enhances the recommendation system, affirming the hypothesis that well-informed attribute selection based on confidence levels is crucial for optimizing collaborative filtering outcomes.

Experiments with varying proportions of high-BAC attributes. In the subsequent experimental sequence, we analyze the impact of varying the proportion of high-BAC attributes included in training, denoted by the parameter α , which ranges from 0 to 1. This parameter represents the percentage of attributes with the highest BAC values that are used during model training. As illustrated in Figure 3, we observe a performance increase as alpha grows, reflecting the incorporation of a greater number of informative attributes with high BAC scores. However, the performance peaks and then diminishes as alpha exceeds a certain saturation point, indicating that the inclusion of too many attributes, particularly those with lower BAC scores, can dilute the model’s effectiveness.

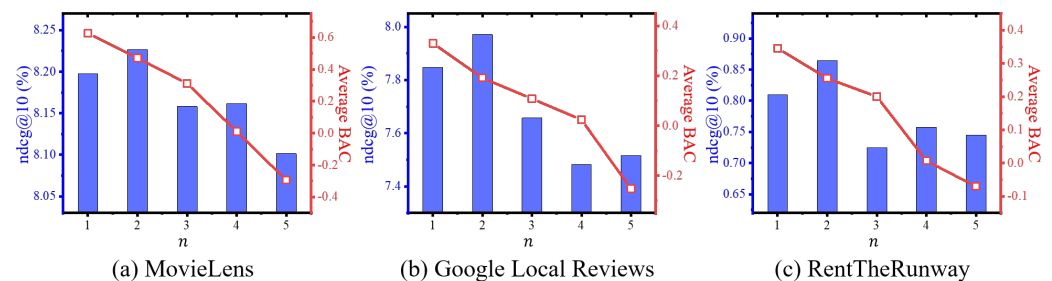


Figure 2. Analysis of performance by BAC quintiles.

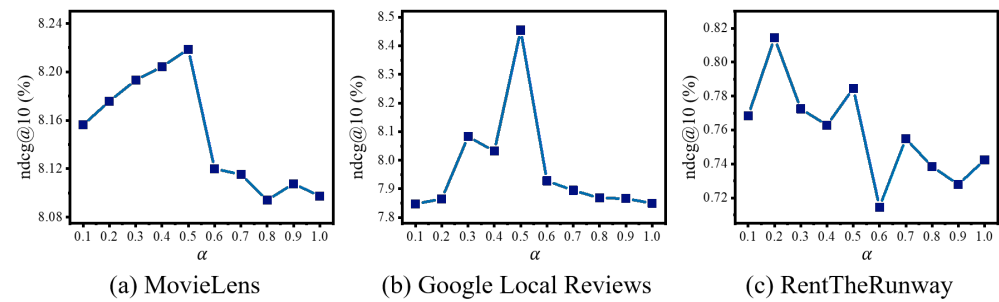


Figure 3. Analysis of performance with varying proportions of high-BAC attributes.

In summary, the outcomes of these two experimental series lend solid support to the effectiveness of our proposed BAC, solidifying its foundational role within our methodology. The precision with which BAC delineates between the most and least informative attributes is crucial, as it underpins the integrity of our subsequent experimental investigations.

5.3. Performance Comparing with Other State-of-the-Art Methods (Q2)

In this section, we present a comparative analysis of the performance between our hypergraph collaborative filtering with attribute inference (HCFA) framework and other cutting-edge methods. As demonstrated in Table 3, HCFA uniformly outperforms competing approaches across all evaluated metrics on three distinct datasets. Specifically, HCFA exceeds the performance of the notable second-best method, LightGCN, by margins of 8.31%, 5.15%, and 13.22% on the respective datasets. While LightGCN stands as a robust baseline with its adept handling of behavior information, its limitations become evident due to its lack of high-order correlation modeling and side information exploitation. These shortcomings are where HCFA gains an edge. Methods that leverage hypergraph structures, such as DHCF and HCCF, indeed tap into high-order information. However, they similarly fall short by not incorporating attribute information, which is a critical component for enriching the recommendation context. Performance metrics for BiANE were suboptimal, primarily because it eschews graph or hypergraph structures in favor of encoding via multiple MLPs. This method significantly underperforms when compared to the others, highlighting the importance of structured modeling in capturing complex user–item interactions. Lastly, AGCN integrates attribute information and employs an attribute inference and update strategy. Nevertheless, its lack of a selective process leads to error propagation and consequent performance decline. Additionally, AGCN’s focus on pairwise relationships without considering higher-order correlations further limits its effectiveness within the scope of our evaluation.

Table 3. Comparison of recommendation performance against state-of-the-art methods. The top-performing metrics are accentuated in **bold** for the best and underlined for the second-best. Performance improvements are quantified by comparing HCFA against the highest-scoring baseline methods.

Models	MovieLens				Google Local Reviews				RentTheRunway			
	Precision	Recall	NDCG	HR	Precision	Recall	NDCG	HR	Precision	Recall	NDCG	HR
NGCF [16]	7.06 $\pm$ 0.03	20.69 $\pm$ 0.24	21.21 $\pm$ 0.23	15.56 $\pm$ 0.16	3.43 $\pm$ 0.27	15.91 $\pm$ 1.93	16.10 $\pm$ 1.94	9.46 $\pm$ 0.91	0.13 $\pm$ 0.04	0.92 $\pm$ 0.29	0.92 $\pm$ 0.29	0.60 $\pm$ 0.18
LightGCN [20]	<u>7.11$\pm$0.06</u>	<u>21.05$\pm$0.10</u>	<u>21.58$\pm$0.11</u>	<u>15.96$\pm$0.08</u>	<u>3.54$\pm$0.15</u>	<u>16.40$\pm$1.09</u>	<u>16.58$\pm$1.08</u>	<u>10.11$\pm$0.31</u>	<u>0.17$\pm$0.03</u>	<u>1.25$\pm$0.20</u>	<u>1.25$\pm$0.20</u>	<u>0.68$\pm$0.05</u>
DHCF [21]	6.79 $\pm$ 0.05	20.42 $\pm$ 0.12	20.94 $\pm$ 0.13	15.19 $\pm$ 0.04	3.10 $\pm$ 0.26	14.84 $\pm$ 2.42	14.97 $\pm$ 2.40	9.33 $\pm$ 1.22	0.12 $\pm$ 0.02	0.89 $\pm$ 0.13	0.89 $\pm$ 0.13	0.49 $\pm$ 0.10
HCCF [22]	6.66 $\pm$ 0.05	19.62 $\pm$ 0.27	20.11 $\pm$ 0.28	14.89 $\pm$ 0.07	2.61 $\pm$ 0.09	12.80 $\pm$ 1.74	12.92 $\pm$ 1.72	7.55 $\pm$ 0.82	0.08 $\pm$ 0.02	0.63 $\pm$ 0.12	0.63 $\pm$ 0.12	0.32 $\pm$ 0.05
AGCN [10]	6.31 $\pm$ 0.01	18.57 $\pm$ 0.23	19.05 $\pm$ 0.23	14.05 $\pm$ 0.12	2.89 $\pm$ 0.19	14.64 $\pm$ 0.41	14.74 $\pm$ 0.44	9.18 $\pm$ 0.43	0.13 $\pm$ 0.02	0.91 $\pm$ 0.22	0.91 $\pm$ 0.22	0.59 $\pm$ 0.08
BiANE [12]	1.24 $\pm$ 0.01	3.51 $\pm$ 0.07	3.62 $\pm$ 0.11	3.64 $\pm$ 0.08	1.06 $\pm$ 0.08	4.22 $\pm$ 0.19	4.28 $\pm$ 0.19	2.44 $\pm$ 0.13	0.03 $\pm$ 0.01	0.24 $\pm$ 0.03	0.24 $\pm$ 0.03	0.14 $\pm$ 0.02
HCFA	8.08$\pm$0.03	22.40$\pm$0.24	22.93$\pm$0.26	17.70$\pm$0.17	3.59$\pm$0.97	17.46$\pm$0.86	17.68$\pm$0.89	10.72$\pm$0.38	0.19$\pm$0.04	1.33$\pm$0.02	1.33$\pm$0.02	0.88$\pm$0.03
Improve	13.67%	6.46%	6.24%	10.95%	1.47%	6.50%	6.62%	6.01%	9.76%	6.88%	6.74%	29.49%

5.4. Ablation Study (Q3)

In this section, we aim to isolate and evaluate the individual contributions of the various components within our HCFA model through a series of ablation studies, as shown in Figure 4. The ablation studies can be categorized into two main parts. The first category pertains to the ablation study of the hypergraph learning module:

- **Hypergraph convolution ablation** removes the hypergraph component, and the model reverts to relying solely on graph convolutions, akin to LightGCN enhanced with attribute inference and updates. We achieve this by setting the hypergraph layer $K = 0$. The full model outperforms this ablation by an average of 2.33%, 28.54%, and 34.14% on three datasets, respectively, underscoring the hypergraph's superior ability to navigate the high-order correlations intrinsic to collaborative filtering with attribute data as opposed to the mere pairwise relationships captured by graph structures.
- **Multi-channel ablation** simplifies the multi-channel framework by merging hypergraphs from different channels into a single channel, as depicted in the equation $H_{A+B} = [H_A, H_B]$. The full model outperforms this ablation by an average of 1.65%, 8.23%, and 18.30% on the three datasets, respectively, validating our hypothesis that attributes and behaviors exert distinct influences on collaborative filtering, and their separate treatment in the model proves to be advantageous for its overall efficacy.
- **Graph convolution ablation** sets the graph convolution layer $J = 0$ to evaluate the impact of the graph convolution layer. The full model outperforms this ablation by an average of 1.76%, 54.47%, and 37.29% on three datasets, respectively, illustrating that pairwise relationships in collaborative filtering are crucial for performance.

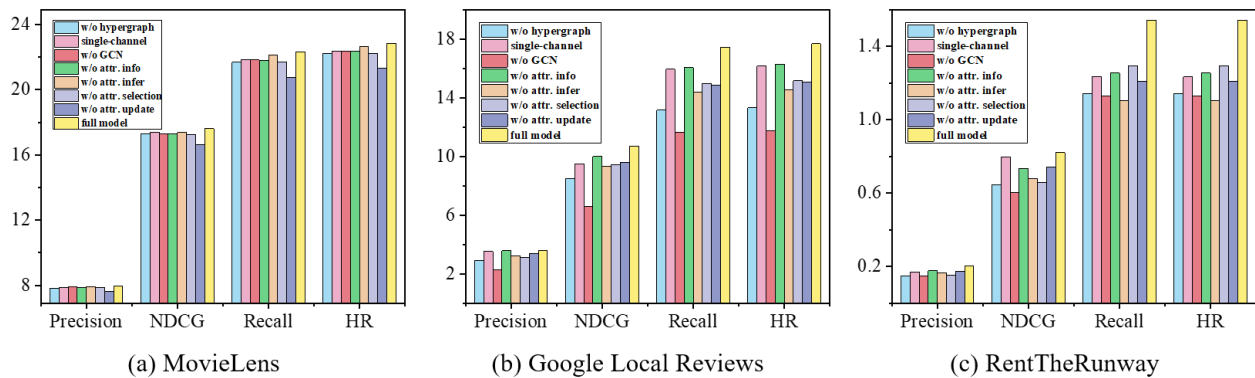


Figure 4. The ablation study of the proposed method.

The second category relates to the utilization of attribute information:

- **Attribute information ablation** assesses the significance of attribute information by removing the attribute channel and the update mechanism, thereby retaining only the behavior channel for model operation. This is implemented by setting $\gamma = 0$ and excluding the attribute hypergraph channel H_A . The full model outperforms this ablation by an average of 1.86%, 6.17%, and 18.30% across the three datasets, respectively. This underscores the critical role that attribute information plays in collaborative filtering, notably in fostering similarities among users and items, a fundamental aspect of the collaborative filtering approach. Interestingly, this ablation model exhibits better performance than subsequent ablations that incorporate attributes, which can be attributed to the high sparsity and imbalance of the attribute data we employ, which, if not handled properly, could detrimentally affect performance. Nevertheless, our HCFA method effectively addresses these issues of sparsity and avoids error propagation, thereby unlocking the full potential and benefits of integrating attribute information into the recommendation process.
- **Attribute inference ablation** seeks to determine the contribution of attribute inference by setting $\gamma = 0$, which means using attribute information solely for constructing

hypergraph structures and not for optimization. The full model outperforms this ablation by an average of 0.91%, 17.20%, and 31.21% across the three datasets, respectively, indicating the effectiveness of attribute inference.

- **Attribute selection ablation** evaluates the effectiveness of the attribute selection module by setting $\beta = 1$, which indicates that all attributes are subject to updates. The full model outperforms this ablation by an average of 0.91%, 17.20%, and 31.21% across the three datasets, respectively, demonstrating the efficacy of selective attribute updating based on the proposed BAC. The validity of BAC is also confirmed in Section 5.2.
- **Attribute update ablation** sets the update number $L = 0$ to assess the impact of the attribute update mechanism. The full model outperforms this ablation by an average of 6.14%, 13.18%, and 20.89% across the three datasets, respectively, indicating the effectiveness of attribute updates informed by the proposed BAC.

5.5. Sensitivity Analysis (Q4)

To investigate the robustness of our model and identify key influencing hyper-parameters, we conduct sensitivity analyses. Firstly, we varied the value of γ from 0.0001 to 0.5, as per Equation (15), with the results displayed in Figure 5. The model demonstrates optimal performance at γ values of 0.01, 0.001, and 0.001 for the three datasets, respectively.

Next, we explore the influence of the hyper-parameter β , which represents the proportion of attributes with high behavior-based attribute confidence (BAC) selected for updates (Section 4.5). We adjusted β from 0.1 to 0.9 and found that the model performance initially increases with more high-BAC attributes included for training. The optimal performance is achieved at β values of 0.8, 0.4, and 0.2 for the respective datasets, as shown in Figure 6. However, performance declines when β becomes too large, incorporating too many attributes with low behavioral relevance, as consistent with observations in Section 5.2.

Finally, we assess the impact of the numbers of hypergraph and graph convolution layers, K and J , ranging from 1 to 5. As shown in Figures 7 and 8, the model performs well across all datasets with K and J set to 1 or 2. However, higher values lead to a decline in performance due to over-smoothing, a phenomenon where vertex features converge and become less distinctive. Therefore, we commonly select 2 as the value for both K and J layer numbers.

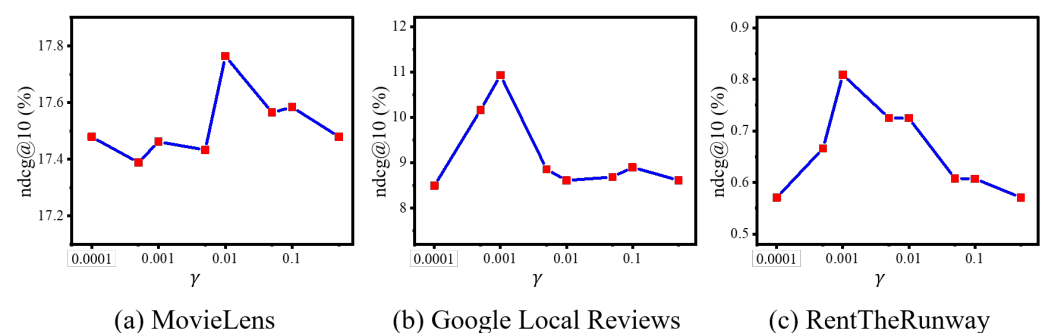


Figure 5. NDCG variations when altering the value of γ in Equation (15) across three datasets.

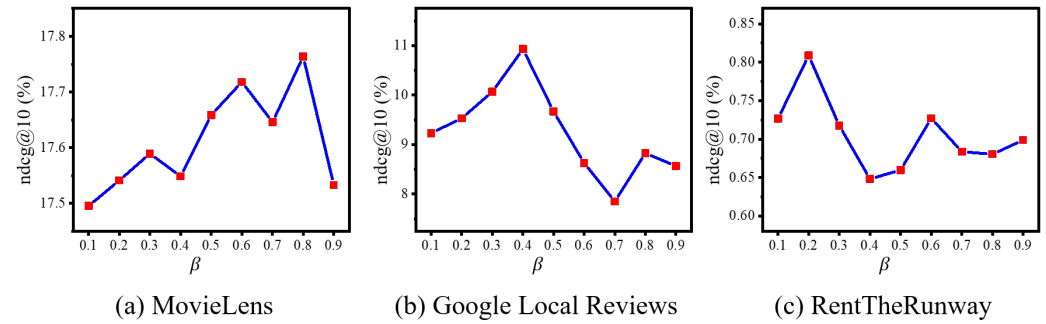


Figure 6. Variations in NDCG when adjusting the value of β across three datasets.

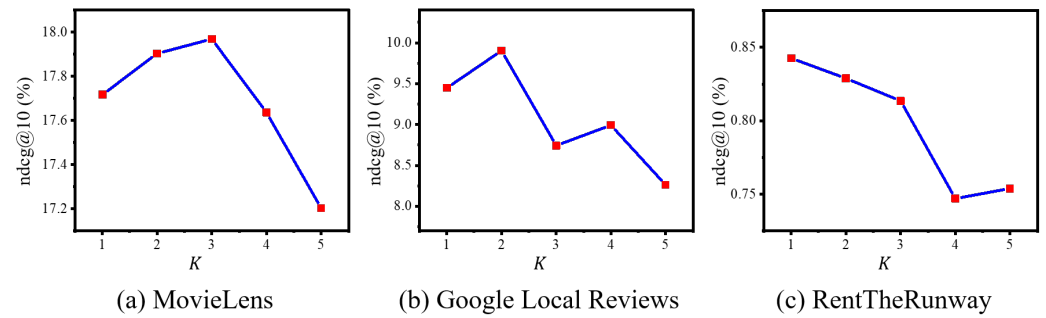


Figure 7. NDCG variations with different numbers of hypergraph convolution layers K across three datasets.

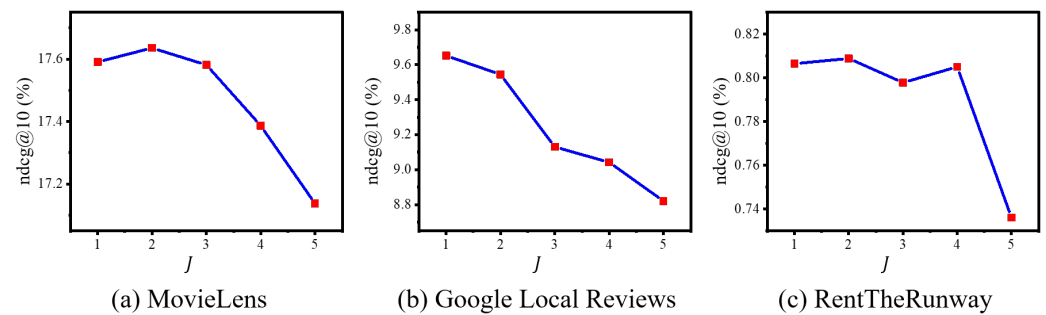


Figure 8. NDCG variations with different numbers of hypergraph convolution layers J across three datasets.

5.6. Interpretable Analysis

In Section 5.2 and the sensitivity analysis of β (Section 5.5, Figure 6), we have already demonstrated the validity of the proposed behavior-based attribute confidence (BAC). Here, we present a visualization of BAC's distribution. "Existing attributes" refer to the ground truth attribute information of users and items, whereas "non-existing attributes" represent attributes not present in the data. For instance, if a user is identified as "Female", then the "Female" attribute is an existing attribute for this user, while "Male" is a non-existing attribute. Similarly, for an item categorized as "Action" and "Sci-Fi", these are its existing attributes, with other categories being non-existing attributes.

As depicted in Figure 9, we generally observe higher BAC scores for existing attributes compared to non-existing ones. This trend validates the rationale behind BAC, suggesting that most existing attributes are indeed reflective of behavior. However, there are instances where some existing attributes display a lower BAC than non-existing ones, indicating that these attributes are relatively less relevant to the corresponding behavior.

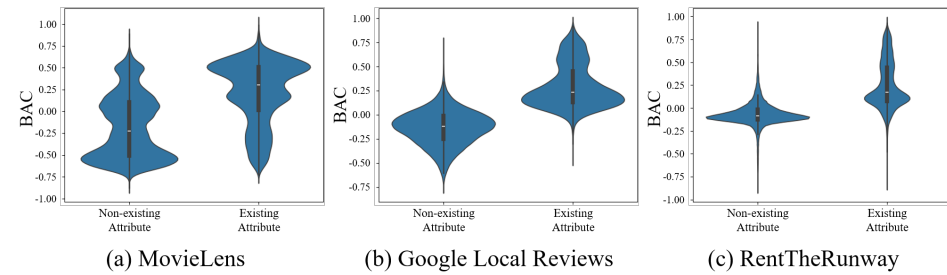


Figure 9. Violin plots depicting the distribution of BAC values for existing and non-existing attributes across three datasets.

6. Conclusions

In conclusion, this study introduces the novel hypergraph collaborative filtering with attribute inference (HCFA) framework, which is specifically tailored for collaborative filtering by integrating attribute information. HCFA stands out for its use of multi-channel hypergraphs to capture high-order correlations from both attributes and behaviors, leading to more refined data modeling. A key innovation of HCFA is the behavior-based attribute confidence (BAC) metric, which quantitatively assesses the credibility of inferred attributes in relation to behaviors. By prioritizing attributes with high BAC scores in subsequent training iterations, HCFA addresses challenges related to attribute sparsity and mitigates the risk of error propagation. Our extensive experimental evaluations confirm the effectiveness of BAC and demonstrate HCFA's superior performance compared to existing methods. Overall, this work contributes to the field by providing a comprehensive framework for collaborative filtering that leverages attribute information to enhance recommendation accuracy and robustness.

Author Contributions: Conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing—original draft preparation, visualization, Y.J.; resources, writing—review and editing, Y.G. and Y.S.; supervision, funding acquisition, C.Y. and S.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Nature Science Foundation of China (61931008, U21B2024, 62071415), the Zhejiang Provincial Natural Science Foundation of China (LDT23F01011F01, LDT23F01015F01, LDT23F01014F01), and the “Pioneer” and “Leading Goose” R&D Program of Zhejiang Province (2022C01068).

Data Availability Statement: The data used to support the findings of this study can be found freely at <https://grouplens.org/datasets/movielens/1m/>, https://cseweb.ucsd.edu/~jmcauley/datasets.html#clothing_fit and https://cseweb.ucsd.edu/~jmcauley/datasets.html#google_local (accessed on 30 October 2023).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

R	User-item interaction matrix
U, V	User and item embedding
P, Q	User and item attribute embedding
X, Y	User and item attribute matrix
A, H	Hypergraph adjacency and incidence matrix
A, B	Attribute and behavior hypergraph
n_u, n_v	User and item number
n_p, n_q	User and item attribute number
d	Embedding dimension
K, J	Hypergraph and graph layer number
L	Update number

References

- Chen, T.; Zhang, W.; Lu, Q.; Chen, K.; Zheng, Z.; Yu, Y. SVDFeature: A Toolkit for Feature-Based Collaborative Filtering. *J. Mach. Learn. Res.* **2012**, *13*, 3619–3622.
- Rendle, S. Factorization Machines with Libfm. *ACM Trans. Intell. Syst. Technol.* **2012**, *3*, 1–22. [\[CrossRef\]](#)
- Gong, N.Z.; Talwalkar, A.; Mackey, L.; Huang, L.; Shin, E.C.R.; Stefanov, E.; Shi, E.; Song, D. Joint Link Prediction and Attribute Inference Using a Social-Attribute Network. *ACM Trans. Intell. Syst. Technol.* **2014**, *5*, 1–20. [\[CrossRef\]](#)
- Yang, C.; Zhong, L.; Li, L.J.; Jie, L. Bi-directional Joint Inference for User Links and Attributes on Large Social Graphs. In Proceedings of the ACM The Web Conference, Perth, Australia, 3–7 April 2017.
- Zhang, H.; Shen, Z. News Recommendation Based on User Topic and Entity Preferences in Historical Behavior. *Information* **2023**, *14*, 60. [\[CrossRef\]](#)
- Giuffrida, G.; Zarba, C.G. A Recommendation Algorithm for Personalized Online News based on Collective Intelligence and Content. In Proceedings of the International Conference on Agents and Artificial Intelligence, Rome, Italy, 28–30 January 2011; SciTePress: Setúbal, Portugal, 2011; pp. 189–194.
- Fischer, E.; Dallmann, A.; Hotho, A. Personalization Through User Attributes for Transformer-Based Sequential Recommendation. In *Workshop on Recommender Systems in Fashion and Retail*; Corona Pampín, H.J., Shirvany, R., Eds.; Springer Nature: Cham, Switzerland, 2023; pp. 25–43.
- Lai, S.; Meng, E.; Zhang, F.; Li, C.; Wang, B.; Sun, A. An Attribute-Driven Mirror Graph Network for Session-based Recommendation. In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–15 July 2022; pp. 1674–1683.
- Anwar, M.M.; Liu, C.; Li, J. Uncovering Attribute-Driven Active Intimate Communities. In *Databases Theory and Applications, 29th Australasian Database Conference, ADC 2018, Gold Coast, QLD, Australia, 24–27 May 2018*; Springer International Publishing: Cham, Switzerland, 2018; pp. 109–122.
- Wu, L.; Yang, Y.; Zhang, K.; Hong, R.; Fu, Y.; Wang, M. Joint Item Recommendation and Attribute Inference: An Adaptive Graph Convolutional Network Approach. In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, 25–30 July 2020; pp. 679–688.
- Rashed, A.; Grabocka, J.; Schmidt-Thieme, L. Attribute-Aware Non-Linear Co-Embeddings of Graph Features. In Proceedings of the ACM Conference on Recommender Systems, Copenhagen, Denmark, 16–20 September 2019; pp. 314–321.
- Huang, W.; Li, Y.; Fang, Y.; Fan, J.; Yang, H. BiANE: Bipartite Attributed Network Embedding. In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, Xi'an, China, 25–30 July 2020; pp. 149–158.
- Qian, T.; Liang, Y.; Li, Q.; Xiong, H. Attribute Graph Neural Networks for Strict Cold Start Recommendation. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 3597–3610. [\[CrossRef\]](#)
- Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
- Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph Attention Networks. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
- Wang, X.; He, X.; Wang, M.; Feng, F.; Chua, T.S. Neural Graph Collaborative Filtering. In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, 21–25 July 2019; pp. 165–174.
- Berg, R.v.d.; Kipf, T.N.; Welling, M. Graph Convolutional Matrix Completion. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Halifax, NS, USA, 13–17 August 2017.
- Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W.L.; Leskovec, J. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 974–983.
- Wu, F.; Zhang, T.; de Souza, A.H., Jr.; Fifty, C.; Yu, T.; Weinberger, K.Q. Simplifying Graph Convolutional Networks. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 10–15 June 2019; Chaudhuri, K., Salakhutdinov, R., Eds.; Proceedings of Machine Learning Research; Volume 97, pp. 6861–6871.
- He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; Wang, M. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, 25–30 July 2020; pp. 639–648.
- Ji, S.; Feng, Y.; Ji, R.; Zhao, X.; Tang, W.; Gao, Y. Dual Channel Hypergraph Collaborative Filtering. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Online, 23–27 August 2020; pp. 2020–2029.
- Xia, L.; Huang, C.; Xu, Y.; Zhao, J.; Yin, D.; Huang, J. Hypergraph Contrastive Collaborative Filtering. In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–12 July 2022; pp. 70–79.
- Yu, J.; Yin, H.; Li, J.; Wang, Q.; Hung, N.Q.V.; Zhang, X. Self-Supervised Multi-Channel Hypergraph Convolutional Network for Social Recommendation. In Proceedings of the Web Conference, Ljubljana, Slovenia, 19–23 April 2021; pp. 413–424.
- Xia, X.; Yin, H.; Yu, J.; Wang, Q.; Cui, L.; Zhang, X. Self-Supervised Hypergraph Convolutional Networks for Session-based Recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtually, 2–9 February 2021; pp. 4503–4511.

25. Xia, L.; Huang, C.; Zhang, C. Self-Supervised Hypergraph Transformer for Recommender Systems. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Washington, DC, USA, 14–18 August 2022; pp. 2100–2109.
26. Li, B.; Zhou, M.; Zhang, S.; Yang, M.; Lian, D.; Huang, Z. BSAL: A Framework of Bi-component Structure and Attribute Learning for Link Prediction. In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–12 July 2022; pp. 2053–2058.
27. Gao, Y.; Zhang, Z.; Lin, H.; Zhao, X.; Du, S.; Zou, C. Hypergraph Learning: Methods and Practices. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *44*, 2548–2566. [[CrossRef](#)] [[PubMed](#)]
28. Feng, Y.; You, H.; Zhang, Z.; Ji, R.; Gao, Y. Hypergraph Neural Networks. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 29–31 January 2019; Volume 33, pp. 3558–3565.
29. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is All You Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
30. Koren, Y.; Bell, R.; Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer* **2009**, *42*, 30–37. [[CrossRef](#)]
31. Harper, F.M.; Konstan, J.A. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* **2015**, *5*, 1–19. [[CrossRef](#)]
32. Misra, R.; Wan, M.; McAuley, J.J. Decomposing fit semantics for product size recommendation in metric spaces. In Proceedings of the ACM Conference on Recommender Systems, Vancouver, BC, Canada, 2–7 October 2018; pp. 422–426.
33. Li, J.; Shang, J.; McAuley, J.J. UCTopic: Unsupervised Contrastive Learning for Phrase Representations and Topic Mining. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Dublin, Ireland, 22–27 May 2022; pp. 6159–6169.
34. Yan, A.; He, Z.; Li, J.; Zhang, T.; McAuley, J.J. Personalized Showcases: Generating Multi-Modal Explanations for Recommendations. In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, Taipei, Taiwan, 23–27 July 2023; pp. 2251–2255.
35. Kingma, D.P.; Ba, J.L. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.