A Robust Approach for Hybrid Personalized Recommender Systems

Le Nguyen Hoai Nam[1,2(✉)]

[1] Faculty of Information Technology, University of Science, Ho Chi Minh City, Vietnam
lnhnam@fit.hcmus.edu.vn
[2] Vietnam National University, Ho Chi Minh City, Vietnam

Abstract. The personalization of services for users is one of the most crucial objectives of digital platforms. This objective is accomplished by integrating automated recommendation components into information systems. The increasing computational power and storage capacity available today have opened up opportunities to deploy a combination of diverse approaches to enhance the accuracy of the recommendation process. Compared to previous research, the distinguishing feature of this study is the introduction of an approach that combines not only computational aspects but also data types. In terms of computation, our approach integrates both item-based and user-based recommendations. Regarding data type, we utilize all three common data types, i.e., user ratings, user reviews, and user interactions, to learn user preferences for recommendations. This comprehensive combination has demonstrated its effectiveness through experiments.

Keywords: Collaborative filtering · recommender systems · personalization

1 Introduction

Recommender systems play a vital role in enterprise information systems. These systems enable users to easily access items that match their interests, facilitating precise and time-saving decision-making [1, 2]. With such benefits, recommendation components have become essential to implementing information systems. Evidence of this can be seen in the significant number of users on digital platforms who make purchases based on recommender systems [3].

To successfully provide recommendations, recommender systems always strive for effective approaches to predict user preferences. Collaborative filtering is one of the most popular approaches for this task, comprising two primary classes: latent factor models and neighbor models. Latent factor models concentrate on representing users and items through latent factors, which enable precise prediction of user preferences for items [4, 5]. However, interpreting the underlying meanings of these latent factors can be challenging. In contrast, neighbor models offer greater interpretability [6–8]. Nowadays, the interpretability of a recommendation model is considered as significant as its accuracy [8, 9].

O. Alonso et al. (Eds.): TPDL 2023, LNCS 14241, pp. 160–172, 2023.
https://doi.org/10.1007/978-3-031-43849-3_14

Neighbor models predict the preference of a user u for an item i by leveraging the preferences for i observed from users who share similar preferences with u, known as neighbor users. In addition to such neighbor-user models, this principle can also be implemented into neighbor-item models. Neighbor-item models entail aggregating the ratings of u for items that are similar to i, referred to as neighbor items. To improve the accuracy of recommendations, many studies have focused on combining the preference predictions from both neighbor-user and neighbor-item models [10, 11]. This paper aims to contribute to the advancement of combined neighbor models, as follows:

- Traditionally, neighbor models rely on observed preferences to identify neighbor users and neighbor items. However, beyond observed preferences, recommender systems can also gather interactions and textual reviews from users [12, 13]. The distinctive feature of this paper is the integration not only of the two computational aspects (neighbor-item model and neighbor-user model) but also of all three popular user profile types (user preferences, user reviews, and user interactions).
- However, the integrations mentioned above may lead to an increase in computational expenses. Hence, our objective is to put forth efficient solutions to address the implementation challenges associated with our proposed approach.

2 Related Works

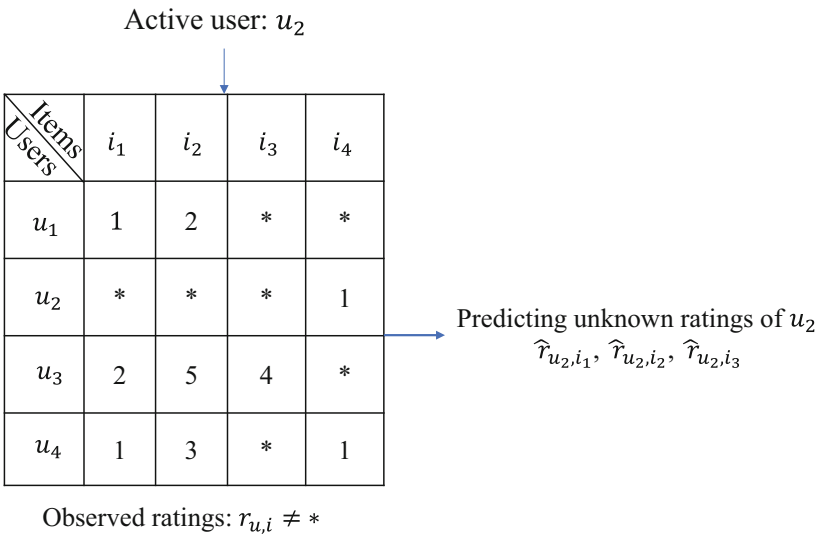

Fig. 1. An description of a recommender system

2.1 Recommendation Problem Statement

Recommendation models are trained based on item preferences observed by users. These preferences typically consist of numeric ratings assigned by users to items ($r_{u,i} \neq *$ where $u = 1 \ldots m$, $i = 1 \ldots n$, m is the number of users, and n is the number of items). The trained models help predict unknown ratings ($r_{u,i} = *$ where $u = 1 \ldots m$ and $i = 1 \ldots n$). In Fig. 1, it is necessary to predict u_2's rating for i_1, i_2, and i_3. The items that receive the highest predicted ratings will be recommended to u_2.

2.2 Recommendation Models

Collaborative filtering is one of the most popular approaches for rating predictions, comprising two primary classes: latent factor models and neighbor models. In the latent factor models, a set of latent factors is learned by optimizing an objective function. Consequently, the rating that a user assigns to an item ($\hat{r}_{u,i}$) can be predicted by multiplying their respective vectors defined by the latent factors ($\overrightarrow{z}_u$ $u = 1 \ldots m$ and $\overrightarrow{h}_i$ $i = 1 \ldots n$) [12–14], as follows:

$$\hat{r}_{u,i} = \overrightarrow{z}_u \cdot \overrightarrow{h}_i \tag{1}$$

The objective functions for learning latent factors are typically constructed based on the principle of optimizing the difference between the observed rating values and the predicted rating values. These objective functions, along with their optimization processes, have demonstrated enhanced effectiveness when incorporating side data [12, 13, 15–17]. For instance, in the study [15], textual descriptions of items are utilized to initialize the variables in the objective function optimization process. Additionally, [12] incorporates user interactions for items to build an objective function for a multiple-step decision-making process. In another study [17], the objective function is modified to optimize the distances between predicted rating values and sentiment values expressed in user textual reviews. Also utilizing reviews, [13] redefines the objective function by incorporating two meanings extracted from reviews: user satisfaction and user experience with items.

The neighbor models are acknowledged for their higher level of interpretability in contrast to the latent factor models [6–8]. Specifically, when making predictions for the rating of a user u on an item i, neighbor models proceed with the following steps [18, 19]:

- Step 1: Identify the group of users ($\mathbb{U}_i$) who have provided ratings for i.
- Step 2: From the users in $\mathbb{U}_i$, establish the neighbor set ($\mathbb{T}_{u,i}$) consisting of users who possess the closest preferences to u.
- Step 3: Compute the average of the observed ratings assigned by the users $v \in \mathbb{T}_{u,i}$ to i ($r_{v,i}$), resulting in an estimation of u's rating for i ($\hat{r}_{u,i}$), as follows:

$$\hat{r}_{u,i} = \mu_u + \frac{\sum_{v \in \mathbb{T}_{u,i}} sim_{u,v}.(r_{v,i} - \mu_v)}{\sum_{v \in \mathbb{T}_{u,i}} |sim_{u,v}|} \tag{2}$$

where μ_u and μ_v denote the averages of the observed ratings of user u and user v, respectively. Step 2 requires the similarity of preferences between user u and each user

v in the set $\mathbb{U}_i$, denoted by $sim_{u,v}$. Some typical methods for calculating similarity are as follows. For Jaccard [20], the more items two users rate in common, the higher their similarity. Going into the details of each rating value, PCC [21] is the cosine of two vectors containing the shared ratings of the two users. MSD [22] uses the absolute difference in shared ratings. [23] calculates the similarity of two users by combining their consistent extreme behaviors and individual extreme behaviors.

In addition to considering observed ratings, observed reviews are also incorporated in the computation of user preference similarity. For example, the authors in [24] calculate the similarity between two users by averaging the distances between their review vectors, which are obtained through a topic modeling technique. In [25], the similarity using reviews is combined with the similarity using ratings. This integration of both rating-based and review-based similarities enhances the accuracy of the similarity calculation.

The aforementioned predictive approaches can also be implemented using a neighbor-item model, where the focus is on neighbor items instead of neighbor users. Accordingly, a user's preference for an item is calculated by aggregating the ratings expressed by the user after experiencing the neighbor items. Several studies have explored rating predictions by combining both the neighbor-user and neighbor-item models. Specifically, [10] employed Singular Value Decomposition on the combined matrix of user numeric ratings and item textual descriptions to derive user/item vectors in the Bert space. Cosine similarity was then calculated for each pair of user/item vectors in the Bert space. These similarity measures were subsequently employed to identify the neighbor users and neighbor items within the combined model. Compared to [10], the difference of [11] lies in the implementation of Singular Value Decomposition on each user/item cluster. Subsequently, the transformed vector of each item/user is utilized to calculate the cosine similarity with other users/items within the same cluster only. Both approaches, [10] and [11], apply the unweighted averaging technique to combine the rating predictions from both neighbor-user and neighbor-item models.

3 Motivation

In addition to ratings, a user's characteristics can also be revealed through his/her interactions in the system, such as clicking, purchasing, or viewing items. Compared to rating data, this data can be collected easily and rapidly through software integrated into the system. After interacting with and rating an item, users often write a review to express their emotions and experiences related to the item. In this paper, we aim to combine not just **T**wo computational **A**spects (neighbor-user model and neighbor-item model) but also **T**hree popular user **P**rofile types (user preferences, user reviews, and user interactions) to enhance neighbor-based recommendation processes. With that idea, we name the proposed approach in this paper **TATP**.

In our recent research, we have introduced two latent factor models: SC1 [12], which combines user interactions and ratings, and UI2R [13], which combines user reviews and ratings. These models have demonstrated remarkable effectiveness in predicting ratings. The integration of rating, interaction, and review data has facilitated the learning of latent factor vectors for more accurate representations of users and items. Building upon this finding, our goal is to leverage these vectors to improve the quality of neighbor users and neighbor items in the TATP. The detailed process is illustrated in Sects. 4.1 and 4.2.

These combinations increase the computational cost of TATP, which significantly impacts the scalability of the system. To address this issue and make TATP more comprehensive, we have proposed an alternative version of TATP that aims to reduce computational expenses. However, it is important to note that reducing computational costs will inevitably lead to a trade-off with the accuracy of rating predictions. In Sect. 4.3, we will provide a specific implementation of TATP.

4 Our Proposed Approach, TATP

4.1 Hybrid Model

Firstly, we employ our previous recommendation models, SC1 [12] and UI2R [13], to derive user and item vectors. SC1 [12] is a latent factor model that integrates user interaction data and rating data. In particular, SC1 captures the steps of a decision-making process in the following manner:

- A user interacts with an item based on compatibility ($\hat{t}_{u,i}$) between the initial user vector ($\overrightarrow{a}_u u = 1 \ldots m$) and the initial item vector ($\overrightarrow{b}_i i = 1 \ldots n$), as follows:

$$\hat{t}_{u,i} = \overrightarrow{a}_u \cdot \overrightarrow{b}_i \tag{3}$$

- Following the interaction on the item, the user engages with it, and ultimately provides a rating ($\hat{r}_{u,i}$) by aligning the final user vector ($\overrightarrow{z}_u u = 1 \ldots m$) with the final item vector ($\overrightarrow{h}_i i = 1..n$), as follows:

$$\hat{r}_{u,i} = \overrightarrow{z}_u \cdot \overrightarrow{h}_i \tag{4}$$

Using the collected interactions and ratings ($t_{u,i} \neq *$ and $r_{u,i} \neq *$ with $u = 1...m$ and i = 1...n), we can estimate both the initial and final user/item vectors. These two optimizations are solved with the constraint that the initial user/item vectors act as the starting values for the corresponding final user/item vectors in the Stochastic Gradient Descent process, as follows:

$$\min_{\substack{\overrightarrow{a}_u,\ \overrightarrow{b}_i \\ u = 1 \ldots m \\ i = 1 \ldots n}} \frac{1}{2} \cdot \sum_{\{(u,i) | u=1...m \wedge i=1...n \wedge t_{u,i} \neq *\}} \left(t_{u,i} - \hat{t}_{u,i}\right)^2$$

$$\Leftrightarrow \min_{\substack{\overrightarrow{a}_u,\ \overrightarrow{b}_i \\ u = 1 \ldots m \\ i = 1 \ldots n}} \frac{1}{2} \cdot \sum_{\{(u,i) | u=1...m \wedge i=1...n \wedge t_{u,i} \neq *\}} \left(t_{u,i} - \overrightarrow{a}_u . \overrightarrow{b}_i\right)^2$$

$$\min_{\substack{\overrightarrow{z}_u,\ \overrightarrow{h}_i \\ u = 1 \ldots m \\ i = 1 \ldots n}} \frac{1}{2} \cdot \sum_{\{(u,i) | u=1...m \wedge i=1...n \wedge r_{u,i} \neq *\}} \left(r_{u,i} - \hat{r}_{u,i}\right)^2$$

$$\Leftrightarrow \min_{\substack{\overrightarrow{z}_u,\ \overrightarrow{h}_i \\ u=1\ldots m \\ i=1\ldots n}} \frac{1}{2}\cdot \sum_{\{(u,i)|u=1\ldots m\wedge i=1\ldots n\wedge r_{u,i}\neq *\}} \left(r_{u,i}-\overrightarrow{z}_u.\overrightarrow{h}_i\right)^2$$

$$Subject\ to: \vec{z}_u^{(0)}=\vec{a}_u\ u=1\ldots m\wedge \vec{h}_i^{(0)}=\vec{b}_i\ i=1\ldots n \tag{5}$$

In contrast to SC1, UI2R [13] is designed to combine textual reviews and numerical ratings in a latent factor model. Each review is encoded into a vector using the Bert model ($\overrightarrow{v}_{u,i}\neq *$). The distinctive feature of this model lies in its consideration of the Bert review vectors as the representation of contextual factors. These factors have a direct influence on the user's rating for the item ($\hat{r}_{u,i}$). This influence is incorporated into the objective function to learn the user vectors ($\overrightarrow{k}_u\ u=1\ldots m$) and item vectors ($\overrightarrow{y}_i\ i=1\ldots n$) in the following manner:

$$\hat{r}_{u,i}\approx \overrightarrow{k}_u.\overrightarrow{y}_i$$

$$\min_{\substack{\overrightarrow{k}_u,\ \overrightarrow{y}_i \\ u=1\ldots m \\ i=1\ldots n}} \frac{1}{2}\cdot \sum_{\{(u,i)|u=1\ldots m\wedge i=1\ldots n\wedge r_{u,i}\neq *\}} \left(r_{u,i}-\hat{r}_{u,i}-\overrightarrow{k}_u.\overrightarrow{v}_{u,i}-\overrightarrow{y}_i.\overrightarrow{v}_{u,i}\right)^2$$

$$\Leftrightarrow \min_{\substack{\overrightarrow{k}_u,\ \overrightarrow{y}_i \\ u=1\ldots m \\ i=1\ldots n}} \frac{1}{2}\cdot \sum_{\{(u,i)|u=1\ldots m\wedge i=1\ldots n\wedge r_{u,i}\neq *\}} \left(r_{u,i}-\hat{r}_{u,i}-\overrightarrow{k}_u.\overrightarrow{v}_{u,i}-\overrightarrow{y}_i.\overrightarrow{v}_{u,i}\right)^2 \tag{6}$$

In the combined latent factor spaces, users and items are represented as specified vectors. Therefore, it is straightforward to implement the cosine similarity to calculate the similarity of two users (u and v)/two items (i and j):

$$sim_{u,v}^{(Rating\&Interaction)}=cosine\left(\overrightarrow{z}_u,\ \overrightarrow{z}_v\right)=\frac{\overrightarrow{z}_u.\overrightarrow{z}_v}{\|\overrightarrow{z}_u\|.\|\overrightarrow{z}_v\|}$$

$$sim_{u,v}^{(Rating\&Review)}=cosine\left(\overrightarrow{k}_u,\ \overrightarrow{k}_v\right)=\frac{\overrightarrow{k}_u.\overrightarrow{k}_v}{\|\overrightarrow{k}_u\|.\|\overrightarrow{k}_v\|}$$

$$sim_{i,j}^{(Rating\&Interaction)}=cosine\left(\overrightarrow{h}_i,\ \overrightarrow{h}_j\right)=\frac{\overrightarrow{h}_i.\overrightarrow{h}_j}{\|\overrightarrow{h}_i\|.\|\overrightarrow{h}_j\|}$$

$$sim_{i,j}^{(Rating\&Review)}=cosine\left(\overrightarrow{y}_i,\ \overrightarrow{y}_j\right)=\frac{\overrightarrow{y}_i.\overrightarrow{y}_j}{\|\overrightarrow{y}_i\|.\|\overrightarrow{y}_j\|} \tag{7}$$

Let the neighbor sets $\mathbb{T}_{u,i}^{(Rating\&Interaction)}$ and $\mathbb{T}_{u,i}^{(Rating\&Review)}$ respectively denote the set of users who have provided ratings for an item i and have the highest similarities ($sim_{u,v}^{(Rating\&Interaction)}$ and $sim_{u,v}^{(Rating\&Review)}$) with a user u. Now, the process of predicting the rating of u for i in the neighbor-user model ($\hat{r}_{u,i}^{(U\_Rating\&Interaction)}$ and $\hat{r}_{u,i}^{(U\_Rating\&Review)}$) would be as follows:

$$\hat{r}_{u,i}^{(U\_Rating\&Interaction)} = \mu_u + \frac{\sum_{v \in \mathbb{T}_{u,i}^{(Rating\&Interaction)}} sim_{u,v}^{(Rating\&Interaction)} .(r_{v,i} - \mu_v)}{\sum_{v \in \mathbb{T}_{u,i}^{(Rating\&Interaction)}} |sim_{u,v}^{(Rating\&Interaction)}|}$$

$$\hat{r}_{u,i}^{(U\_Rating\&Review)} = \mu_u + \frac{\sum_{v \in \mathbb{T}_{u,i}^{(Rating\&Review)}} sim_{u,v}^{(Rating\&Review)} .\left(r_{v,i} - \mu_v\right)}{\sum_{\mathbb{T}_{u,i}^{(Rating\&Review)}} \left|sim_{u,v}^{(Rating\&Review)}\right|} \quad (8)$$

Similarly, the prediction process in the neighbor-item model ($\hat{r}_{u,i}^{(I\_Rating\&Interaction)}$ and $\hat{r}_{u,i}^{(I\_Rating\&Review)}$) is implemented as follows:

$$\hat{r}_{u,i}^{(I\_Rating\&Interaction)} = \mu_u + \frac{\sum_{j \in \mathbb{W}_{i,u}^{(Rating\&Interaction)}} sim_{i,j}^{(Rating\&Interaction)} .(r_{u,j} - \mu_u)}{\sum_{j \in \mathbb{W}_{i,u}^{(Rating\&Interaction)}} |sim_{i,j}^{(Rating\&Interaction)}|}$$

$$\hat{r}_{u,i}^{(I\_Rating\&Review)} = \mu_u + \frac{\sum_{j \in \mathbb{W}_{i,u}^{(Rating\&Review)}} sim_{i,j}^{(Rating\&Review)} .\left(r_{u,j} - \mu_u\right)}{\sum_{j \in \mathbb{W}_{i,u}^{(Rating\&Review)}} \left|sim_{i,j}^{(Rating\&Review)}\right|} \quad (9)$$

where $\mathbb{W}_{i,u}^{(Rating\&Interaction)}$ and $\mathbb{W}_{i,u}^{(Rating\&Review)}$ represent the sets of items that have been rated by user u and have the highest similarities ($sim_{i,j}^{(Rating\&Interaction)}$ and $sim_{i,j}^{(Rating\&Review)}$) with i.

As mentioned in Sect. 3, we aim for a comprehensive hybrid approach to rating predictions. The comprehensiveness lies in not only combining both user-based and item-based implementations but also incorporating interaction, review, and rating data. Therefore, we utilize the weighted average to achieve the final rating ($\hat{r}_{u,i}$), as follows:

$$\hat{r}_{u,i} = \alpha.\hat{r}_{u,i}^{(U\_Rating\&Interaction)} + \beta.\hat{r}_{u,i}^{(U\_Rating\&Review)} + \gamma.\hat{r}_{u,i}^{(I\_Rating\&Interaction)} + \sigma.\hat{r}_{u,i}^{(I\_Rating\&Review)} \quad (10)$$

4.2 Weight Estimation

Many previous hybrid models often assign equal weights to their individual models ($\alpha = \beta = \gamma = \sigma$ in Eq. (10)). However, in reality, the individual models can have varying levels of accuracy in different application domains. Therefore, setting equal weights may not be the optimal choice. In this paper, the weights (α, β, γ, and σ) are determined through an estimation process using a subset of observed data called the validation set

$\mathbb{H}$. Specifically, we use Eq. (8, 9) to make predictions for each rating in the validation set, and then aggregate them using Eq. (10). Optimizing the difference between the observed ratings in the validation set ($r_{u,i} \in \mathbb{H}$) and their predictions ($\hat{r}_{u,i}$) helps determine α, β, γ, and σ. Our advantage lies in the parallel optimization towards both the observed rating and the inferred ratings from observed reviews (the rating $r\prime_{u,i}$ inferred from the review of user u for item i, as proposed in [17]). This parallel optimization is particularly effective in situations where there is an inconsistency between the reviews and the ratings provided by the users. The detailed objective function for weight estimation is as follows:

$$\begin{aligned} &\min_{\alpha,\beta,\gamma,\sigma} \frac{1}{2}. \sum_{r_{u,i}\in\mathbb{H}} \left(\left(r_{u,i} - \hat{r}_{u,i}\right)^2 + \left(r\prime_{u,i} - \hat{r}_{u,i}\right)^2 \right) + \frac{\lambda}{2}\left(\alpha^2 + \beta^2 + \gamma^2 + \sigma^2\right) \\ \Leftrightarrow &\min_{\alpha,\beta,\gamma,\sigma} \frac{1}{2}. \sum_{r_{u,i}\in\mathbb{H}} \left(\begin{array}{l} \left(\begin{array}{c} r_{u,i} - \alpha.\hat{r}_{u,i}^{(U\_Rating\&Interaction)} - \beta.\hat{r}_{u,i}^{(U\_Rating\&Review)} \\ -\gamma.\hat{r}_{u,i}^{(I\_Rating\&Interaction)} - \sigma.\hat{r}_{u,i}^{(I\_Rating\&Review)} \end{array} \right)^2 \\ + \left(\begin{array}{c} r'_{u,i} - \alpha.\hat{r}_{u,i}^{(U\_Rating\&Interaction)} - \beta.\hat{r}_{u,i}^{(U\_Rating\&Review)} \\ -\gamma.\hat{r}_{u,i}^{(I\_Rating\&Interaction)} - \sigma.\hat{r}_{u,i}^{(I\_Rating\&Review)} \end{array} \right)^2 \end{array} \right) + \frac{\lambda}{2}\left(\alpha^2 + \beta^2 + \gamma^2 + \sigma^2\right) \end{aligned} \tag{11}$$

The last part in Eq. (11) is a Tikhonov regularization to prevent overfitting with the weight λ. Equation (11) can be solved as a bridge regression.

4.3 Efficient Implementation

In general, the implementation of a neighbor model consists of two stages: offline and online.

- During the offline stage, the model computes the similarity between each pair of users/items using a selected similarity metric.
- The online stage is performed based on the sets of neighbor users/items, which are easily determined by the similarity scores calculated in the offline stage. Using these neighbor sets, the model predicts the ratings of the active user for items that he/she has not yet discovered.

However, in scenarios where the number of users/items is large, calculating pairwise similarities for all user/item pairs in the offline stage becomes computationally infeasible. One approach to address this issue is to cluster users/items, allowing for similarity calculations only within each cluster [18, 19]. However, clustering users/items in the sparse space of preferences often leads to suboptimal clustering results. This can result in a significant decline in the performance of subsequent neighbor-based recommendations. It is important to highlight that in the offline stage, we have successfully obtained user vectors, i.e., $\overrightarrow{z}_u$ and $\overrightarrow{k}_u$ $u = 1 \ldots m$, and item vectors, i.e., $\overrightarrow{h}_i$ and $\overrightarrow{y}_i$ $i = 1 \ldots n$, in a combined space of ratings, reviews, and interactions. We apply concatenation to these vectors to create a unique vector for each item and user, i.e., $\overrightarrow{x}_u$=concatenation($\overrightarrow{z}_u$, $\overrightarrow{k}_u$) $u = 1 \ldots m$ and $\overrightarrow{d}_i$= concatenation($\overrightarrow{h}_i$, $\overrightarrow{y}_i$) $i = 1 \ldots n$. This technique is commonly used in previous research to integrate different representations of an object. Instead of

clustering sparse preference vectors, we cluster the concatenated vectors, as follows:

$$\begin{aligned} &\text{Clustering } \vec{x}_u \; u = 1...m\,; \;\; \text{clustering } \vec{d}_i \; i = 1...n \\ &sim_{u,v} \text{ if } u \text{ and } v \text{ belong to the same cluster.} \\ &sim_{i,j} \text{ if } i \text{ and } j \text{ belong to the same cluster} \end{aligned} \tag{12}$$

5 Experiment

5.1 Experiment Setup

In the experiments, our method will be compared with the following approaches:

- [23]: Neighbor-User model relying solely on Ratings (NuRa)
- [24]: Neighbor-User model relying solely on Reviews (NuRe)
- [25]: Neighbor-User model combining Ratings and Reviews (NuRaRe)
- [11]: Neighbor-User and Neighbor-Item model relying solely on Ratings (NuNiRa)
- [10]: Neighbor-User and Neighbor-Item model combining Ratings and Reviews (NuNiRaRe) where the item description is formed by aggregating reviews.
- Our proposed approach: Neighbor-User and Neighbor-Item model combining Ratings, Reviews, and Interactions (TATP).

The parameters for the latent factor models SC1 and UI2R to learn user/item vectors in TATP are set as follows:

- Learning rate is 0,003
- Regularization weight is 0,02
- The number of latent factors is 60

5.2 Dataset

To conduct the experiments, we selected three popular Amazon datasets containing both ratings and reviews. Their details are presented in Table 1. The experimental datasets are randomly divided into 65% for training and 25% for testing.

Table 1. The datasets.

	# users	# items	# ratings and reviews
Video games	24,303	10,672	231,780
Clothing-Accessories	39,387	23,033	278,677
Gourmet food	14,681	8,713	151,254

Inspired by [12, 26, 27], we simulate user interaction data as follows. If a user provides a rating for an item, it can be inferred that the user has interacted with the item. This simulation does have a limitation, as it overlooks instances where users interact with items but do not provide ratings. However, given the vast number of items available, omitting these cases introduces negligible bias to the overall results.

5.3 Measure

RMSE is utilized to assess the accuracy of recommendation models, as follows:

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in\mathbb{T}}\left(\hat{r}_{u,i} - r_{u,i}\right)^2}{|\mathbb{T}|}} \tag{13}$$

where $\mathbb{T}$ is the test set.

5.4 Experimental Results

Figure 2 illustrates the RMSE results of the experimental approaches. It is evident that the hybrid approaches, i.e., NuRaRe, NuNiRa, NuNiRaRe, and TATP, yield better results compared to the individual approaches, i.e., NuRe and NuRa. Among the hybrid approaches, our approach TATP outperforms the others. The reason for this is

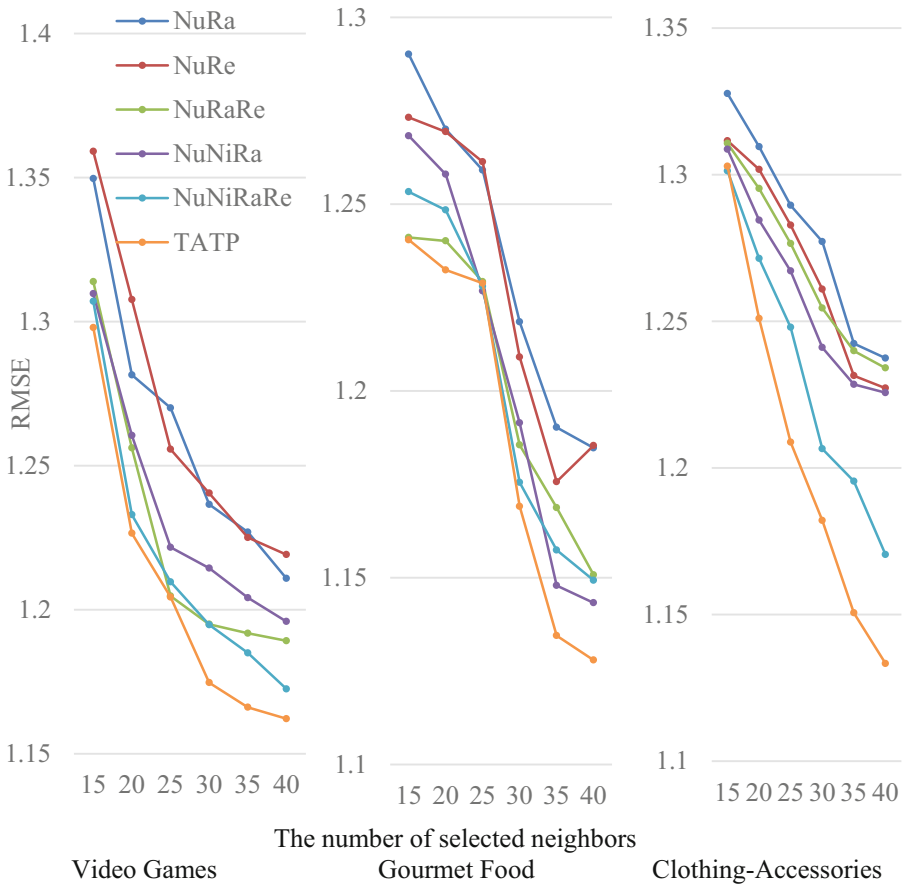

Fig. 2. The RMSE results when the number of selected neighbors increases from 15 to 40.

that TATP utilizes more information and performs more comprehensive calculations compared to the other approaches. Furthermore, leveraging both the observed ratings and observed reviews to implement a bridge regression for learning the weights of the individual models also proves to be highly effective. Due to the inherent sparsity of the recommendation problem, combining more data and computations naturally leads to significantly improved accuracy in the recommendation process. Although there is a trade-off in terms of increased computational costs, the continual advancements in computational power and storage capacity have made this trade-off more feasible and acceptable.

Fixing the number of neighbors optimally for each approach on each experimental dataset, we perform user/item clustering to enhance scalability as presented in Sect. 4.3. Figure 3 shows that the clustering in our combined space (TATP + LatentVectorClus) proves to be more effective compared to that in the preference space [18, 19] (TATP + PreferenceClus). This effect becomes more pronounced as the number of clusters increases. Note that as the number of clusters increases, the number of users/items within each cluster gradually decreases. This means that the number of pairs requiring similarity calculations and the computational cost for neighbor determination decreases as well. As a result, the system scalability is greatly enhanced.

Finally, we conducted RMSE measurements at the individual user level instead of the overall system level. Consequently, for each approaches, we obtained 78,371 RMSE results corresponding to 78,371 users across all three experimental datasets. These sample sets were then subjected to the statistical Wilcoxon signed-ranks test. The advantage of the Wilcoxon signed-ranks test is that it does not require the sample sets to adhere to a normal distribution. As depicted in Table 2, the statistical results demonstrate that our approach TATP significantly outperform other methods in terms of statistical significance, as all the obtained p-values are less than 0.05.

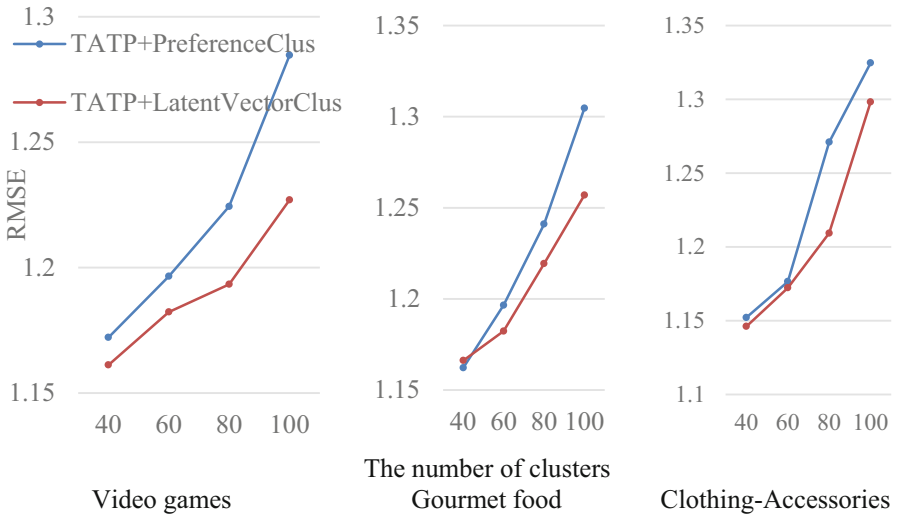

Fig. 3. The RMSE results when the number of clusters increases from 40 to 100.

Table 2. The results of the statistical Wilcoxon signed-ranks test.

TATP >> NuNiRaRe p-value = 0.0152	TATP >> NuNiRa p-value = 0.0067	TATP >> NuRaRe p-value = 0.0081	TATP >> NuRe p-value = 0.0029

6 Conclusion and Future Work

In terms of computation, this paper combines both user and item aspects to effectively utilize both neighbor users and neighbor items in predicting ratings. Regarding data, this paper integrates three common types of user profiles, including user ratings, user reviews, and user interactions, into the training process. The parameters of our hybrid model are estimated using a bridge regression. The experimental results on various datasets demonstrate that our approach performs better than both individual approaches and other hybrid approaches.

The main drawback of our proposed approach is its substantial computational overhead. This arises from the implementation of diverse individual approaches on various user profile types. Although we have proposed a version to reduce computational expenses, the reduction is not significant. Therefore, in the future, we aim to redesign the proposed approach with parallel processing. This will facilitate successful deployment on a distributed Hadoop.

Acknowledgments. This research is funded by the University of Science, VNU-HCM under grant number CNTT 2022-01.

References

1. Duan, R., Jiang, C., Jain, H.K.: Combining review-based collaborative filtering and matrix factorization: a solution to rating's sparsity problem. Decis. Support Syst. **156**, 113748 (2022)
2. Ahmadian, M., Ahmadi, M., Ahmadian, S.: A reliable deep representation learning to improve trust-aware recommendation systems. Expert Syst. Appl. **197**, 116697 (2022)
3. Ahmadian, S., Ahmadian, M., Jalili, M.: A deep learning based trust-and tag-aware recommender system. Neurocomputing **488**, 557–571 (2022)
4. Nam, L.N.H.: Towards comprehensive profile aggregation methods for group recommendation based on the latent factor model. Expert Syst. Appl. **185** (2021)
5. Nam, L.N.H.: Profile aggregation-based group recommender systems: moving from item preference profiles to deep profiles. IEEE Access **10**, 6218–6245 (2022)
6. Noshad, Z., Bouyer, A., Noshad, M.: Mutual information-based recommender system using autoencoder. Appl. Soft Comput. **109**, 107547 (2021)
7. Sun, X., Zhang, L.: Multi-order nearest neighbor prediction for recommendation systems. Digital Signal Process. **127**, 103540 (2022)
8. Nam, L.N.H.: Towards comprehensive approaches for the rating prediction phase in memory-based collaborative filtering recommender systems. Inf. Sci. **589** (2022)
9. Vultureanu-Albişi, A., Bădică, C.: A survey on effects of adding explanations to recommender systems (2022)

10. Hoang, B.N.M., Vy, H.T.H., Hong, T.G., Hang, V.T.M., Nhung, H.L.T.K., Nam, L.N.H.: Using Bert Embedding to improve memory-based collaborative filtering recommender systems. In: 2021 RIVF International Conference on Computing and Communication Technologies (RIVF), pp. 1–6. IEEE (2021)
11. Nilashi, M., Ibrahim, O., Bagherifard, K.: A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques. Expert Syst. Appl. **92**, 507–520 (2018)
12. Nam, L.N.H.: Latent factor recommendation models for integrating explicit and implicit preferences in a multi-step decision-making process. Expert Syst. Appl. **174** (2021)
13. Nam, L.N.H.: Incorporating textual reviews in the learning of latent factors for recommender systems. Electron. Commerce Rese. Appl. **52** (2022)
14. Noulapeu Ngaffo, A., Choukair, Z.: A deep neural network-based collaborative filtering using a matrix factorization with a twofold regularization. Neural Comput. Appl. **34**(9), 6991–7003 (2022)
15. Khan, Z., Iltaf, N., Afzal, H., Abbas, H.: Enriching non-negative matrix factorization with contextual embeddings for recommender systems. Neurocomputing **380**, 246–258 (2020)
16. Yengikand, A.K., Meghdadi, M., Ahmadian, S.: DHSIRS: a novel deep hybrid side information-based recommender system. Multimedia Tools Appl. 1–27 (2023)
17. Shen, R.P., Zhang, H.R., Yu, H., Min, F.: Sentiment based matrix factorization with reliability for recommendation. Expert Syst. Appl. **135**, 249–258 (2019)
18. Nikolakopoulos, A.N., Ning, X., Desrosiers, C., Karypis, G.: Trust your neighbors: a comprehensive survey of neighborhood-based methods for recommender systems. Recommender Systems Handbook, pp. 39–89 (2021)
19. Aggarwal, C.C.: Neighborhood-based collaborative filtering. Recommender Systems: The Textbook, pp. 29–70 (2016)
20. Koutrika, G., Bercovitz, B., Garcia-Molina, H.F.: Expressing and combining flexible recommendations. In: Proceedings of the 35th SIGMOD International Conference on Management of Data (SIGMOD 2009), Providence, RI, USA, vol. 29 (2009)
21. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: ACM SIGIR Forum, vol. 51, no. 2, pp. 227–234. New York, NY, USA: ACM (2017)
22. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv. Artif. Intell. **2009** (2009)
23. Feng, C., Liang, J., Song, P., Wang, Z.: A fusion collaborative filtering method for sparse data in recommender systems. Inf. Sci. **521**, 365–379 (2020)
24. Musto, C., de Gemmis, M., Semeraro, G., Lops, P.: A multi-criteria recommender system exploiting aspect-based sentiment analysis of users' reviews. In Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 321–325 (2017)
25. Ghasemi, N., Momtazi, S.: Neural text similarity of user reviews for improving collaborative filtering recommender systems. Electron. Commer. Res. Appl. **45**, 101019 (2021)
26. Aggarwal, C.C.: An introduction to recommender systems. In: Recommender Systems, pp. 1–28. Springer, Cham (2016)
27. Pan, R., Scholz, M.: Mind the gaps: weighting the unknown in large-scale one-class collaborative filtering. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 667–676 (2009)