A Two-tier Shared Embedding Method for Review-based Recommender Systems

Zhen Yang
yangzhen@bjut.edu.cn
Faculty of Information Technology,
Beijing University of Technology
Beijing, China

Junrui Liu
liujunrui@emails.bjut.edu.cn
Faculty of Information Technology,
Beijing University of Technology
Beijing, China

Tong Li*
litong@bjut.edu.cn
Faculty of Information Technology,
Beijing University of Technology
Beijing, China

Di Wu
wuxiaodou@emails.bjut.edu.cn
Faculty of Information Technology,
Beijing University of Technology
Beijing, China

Shiqiu Yang
ysqbjut@emails.bjut.edu.cn
Beijing-Dublin International College,
Beijing University of Technology
Beijing, China

Huan Liu
huanliu@asu.edu
Computer Science and Engineering,
Arizona State University
Tempe, AZ, USA

ABSTRACT

Reviews are valuable resources that have been widely researched and used to improve the quality of recommendation services. Recent methods use multiple full embedding layers to model various levels of individual preferences,increasing the risk of the data sparsity issue. Although it is a potential way to deal with this issue that models homophily among users who have similar behaviors, the existing approaches are implemented in a coarse-grained way. They calculate user similarities by considering the homophily in their global behaviors but ignore their local behaviors under a specific context. In this paper, we propose a two-tier shared embedding model (TSE), which fuses coarse- and fine-grained ways of modeling homophily. It considers global behaviors to model homophily in a coarse-grained way, and the high-level feature in the process of each user-item interaction to model homophily in a fine-grained way. TSE designs a whole-to-part principle-based process to fuse these ways in the review-based recommendation. Experiments on five real-world datasets demonstrate that TSE significantly outperforms state-of-the-art models. It outperforms the best baseline by 20.50% on the root-mean-square error (RMSE) and 23.96% on the mean absolute error (MAE), respectively. The source code is available at https://github.com/dianziliu/TSE.git.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Collaborative filtering**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Recommender Systems, Review-based recommendation, Shared embedding

*Corresponding author.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom.

ACM ISBN 979-8-4007-0124-5/23/10.
https://doi.org/10.1145/3583780.3614770

ACM Reference Format:
Zhen Yang, Junrui Liu, Tong Li, Di Wu, Shiqiu Yang, and Huan Liu. 2023. A Two-tier Shared Embedding Method for Review-based Recommender Systems. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3583780.3614770

1 INTRODUCTION

Recommendation systems have extensively integrated into web services to improve user experiences [6, 43]. Reviews are important user-generated content, reflect users' sentiments and interests, and have attracted the attention of many researchers [24, 28, 45].

Recently, an increasing number of review-based methods use deep neural networks to learn multiple features and latent representations. They use multiple full embedding layers to model various individual preferences from reviews, click data, and ratings [13, 27, 44]. These methods ignore the sparsity of recommendation data that affects the quality of embedding [17, 37, 49]. Sparse data are far from sufficient to accurately learn individual preferences using multiple full embedding layers. Homophily regularization considers user homophily to address the data sparsity problem [3, 37]. In these works, similar users are restrictively represented by similar embedding vectors, making these vectors more discriminating than the standard full embedding vectors.

However, the existing methods model homophily in a coarse-grained way, which is insufficient for precisely profiling users. In some cases, a user's individual behavior may deviate from their general behavior, but is similar to other users' actions in the same context [2, 12, 47]. Figure 1 gives an example of this phenomenon. As a pop music lover, Tom likes listening to pop music, and thus there is a homophily between Tom and pop music lovers. Generally, pop music lovers dislike classical music, which leads to coarse-grained ways hard to recommend classical music to Tom, but Tom likes to listen to a little classical music sometimes, similar to classical music lovers in some contexts. Coarse-grained ways ignore the individuality of each action and thus make error predictions sometimes. Therefore, it is necessary to model homophily in a fine-grained way that considers the context of user actions.

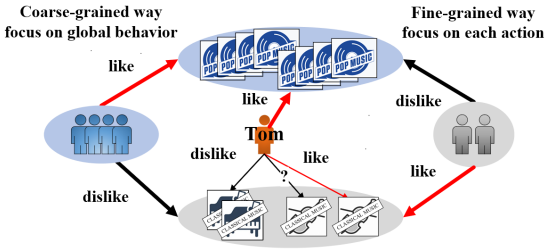


Figure 1: An example of coarse- and fine-grained ways of modeling homophily.

In this paper, we propose a Two-tier Shared Embedding model (TSE) for the review-based recommendation, which fuses coarse- and fine-grained ways of modeling homophily, accurately modeling user behavior to improve model performance. Specifically, TSE consists of a statically shared embedding network and a dynamically shared embedding network, that are used to model coarse- and fine-grained homophily, respectively. The statically shared embedding network shares embedding vectors to model homophily through analysis of global behaviors. Shared embedding uses the same vector to represent similar users, directly using similar users' data to train one vector. Thus, data from similar users can complement each other and be used to train the vector, making the vector generalizes better than a single vector representing users independently. The dynamically shared embedding network models the context of each action by extracting dynamically high-level features and then using it to select the most appropriate fine-grained shared embedding. Because there are some dynamically high-level features in the process of user-item interaction, explaining the reason for interactions, and similar users could have similar high-level features in an interaction. At the same time, considering the diversity of user interests [21], we design two strategies to deal with user interests in different situations, A hard strategy finds the most similar embedding, and a soft strategy models situations where users are influenced by multiple factors when making decisions. To combine with the review-based recommendation, TSE designs a whole-to-part principle-based process to fuse coarse- and fine-grained ways of modeling homophily. It gets coarse-grained shared embedding, combines coarse-grained shared embedding and model input to model fine-grained shared embedding, and combines fine-grained shared embedding and model input to make predictions.

The main contributions of this paper are as follows:

- We propose a two-tier shared embedding method for the review-based recommendation, which fuses coarse- and fine-grained ways of modeling homophily, accurately modeling user preference to improve model performance.
- We propose a dynamically shared embedding network, which models homophily in a fine-grained way by considering dynamically high-level features in user-item interactions.
- Experiments on five real-world datasets demonstrate that TSE significantly outperforms state-of-the-art models. TSE achieves an average improvement of 20.50% in RMSE and 23.96% in MAE.

2 RELATED WORK

2.1 Review-based methods

Reviews are typical features in recommender systems. They have a strong intrinsic correlation with user interests. Review-based methods learn individual preferences by full embedding and make predictions by combining NLP methods, such as LDA in CTR [40], SDAE in CDL [41]. ConvMF [19] uses convolutional neural network (CNN) to extract local semantic features from reviews. These methods extract textual features as a vector. The vector is used as the item latent features in a matrix factorization (MF).

Some researchers have performed complex modeling of reviews. CAPR [22] and ARPM [20] perform aspect and sentiment analysis on textual reviews and then establishes users' and items' preference feature vectors. A2SPR [16] calculates item relevance by an item graph that edges are the number of co-reviewer. RGNN [28] is a type-aware graph attention network that builds a review graph for each user where nodes are words and edges are word orders. MRCP [25] extracts word-level, review-level, and aspect-level features to represent users and items via a three-tier attention network. SENGR [36] is a sentiment-enhanced neural graph method that incorporates the information derived from textual reviews and bipartite graphs.

Modeling interaction behavior is a way of improving model performance. D-attn [34] uses dual local and global attention to model word-level and review-level features. As global attention is applied to both the user side and the item side, it learns the interaction features between the two sides. NARRE [5] filters useless reviews by using the vector representing each user and item as a part of attention scores. DAML [24] employs the local and mutual attention of CNN to learn features from reviews, and then integrates them with the latent factor model for rating prediction. HTI [44] captures interactions based on reviews by mutually propagating textual features at word and review levels and dynamically identifying their importance. NRCA [26] points out two main paradigms of reviews, i.e., the document level and the review level. It uses a cross-attention mechanism to aggregate the informative words and reviews and represent users. TAERT [13] uses three attention networks to model different features, i.e., word contribution, review usefulness, and latent factors. It designs a temporal convolutional network for learning sequential characteristics and interactions of neighboring features and global features. DSRLN [27] extracts static and dynamic user interests and by stacking attention layers that deal with sequence features and attention encoding layers that deal with of user-item interaction.

Recent methods use multiple full embedding layers. For example, HTI has two full embedding layers to determine word- and rating-level preferences; TAERT has one full embedding layer and a transfer matrix to calculate rating-level preferences and review-level attention scores; and DSRLN has one full embedding layer but multiple transfer matrices in various layers to transfer embedding vectors. TSE uses three shared embedding layers to model user preferences in reviews, interactions, and ratings. Shares embedding vectors among similar users improves the training degree of various levels of embedding vectors, and holds great potential in model complexity and performance.

2.2 Cluster-based methods

Cluster-based (CB) methods cluster users and items based on behaviors [39]. They aggregate all group members' interests into a common model [30, 31]. The group representative is then used to recommend to individual users [18]. There are two types, i.e., group preference aggregation(GPA) and individual preference aggregation(IPA) [15].

The main research of GPA methods is to design a suitable clustering method to cluster all users into some groups. The recommendation results are calculated based on each individual group. [31] clusters the tail items while the head items on the ratings are predicted individually. An adaptive clustering (AC) method [30] pre-specified the size of each group by a fixed popularity threshold, which does not cluster head items whose popularity is greater than a threshold. Some researchers tend to use extra information, such as trust relationships, to cluster users [11, 29]. The final recommendation result comes from a support vector regression model, which selects the multiple clustering results. 2D-GC [35] is a 2D-graph clustering method that expands user neighborhoods and searches for the optimal number of groups. [8] considers the asymmetric of items and proposes a novel K-medoids clustering recommendation algorithm based on a probability distribution for collaborative filtering.

IPA methods aggregate the individual results from group members. [47] proposes that users (item) can join multiple clusters (subgroups) and represents the problem as a Multiclass Co-Clustering problem. It proposes a novel loss function resembling graph regularization to find meaningful subgroups. The final prediction score is the max value of prediction scores on all subgroups. DLGR [15] argues that the group influences user choices, i.e., user characteristics are lower-level forms of group characteristics. It includes a dual-wing restricted Boltzmann machine on the top of the deep belief network to learn group features. [18] blends collaborative filtering and content-based recommendation results to produce the final group recommendation results. To aggregate collaboration results, it uses an additive form to obtain a total score for each item. The content-based item score depends on how many users the item was recommended to in a group. The final score is derived from the product of these two scores. SCoC [23] agrees that users' interests are diverse, and a user may belong to multiple groups together. SCoC first clusters users and items into subgroups, and then aggregates the prediction results of the subgroups. The probabilities of user grouping are used as weights for the subgroup results in the aggregation process.

3 METHOD

The framework of TSE is illustrated in Figure 2. Based on existing works [13, 27], TSE utilizes a three-stages to make predictions, including review feature learning, interaction feature learning based on review features, and rating predictions. In each stage, TSE models homophily both in coarse- and fine-grained ways. For briefness, we summarize them into two networks, i.e., a dynamically shared embedding network (DSEN) models the fine-grained homophily, and a statically shared embedding network (SSEN) models the coarse-grained homophily.

3.1 Problem of Formalization

In a recommendation problem, assume that a user set $\mathcal{U} = \{u^1, u^2, \cdots, u^m\}$ and an item set $\mathcal{I} = \{i^1, i^2, \cdots, i^n\}$ contains m users and n items, respectively. r_{ui} denotes the true rating of a user u of an item i, which is one element of rating matrix $R \in \mathbb{R}^{m*n}$. We denote $\mathcal{D}_u = \{d_{u,1}, d_{u,2}, \cdots, d_{u,z_u}\}$ as the set of user reviews and $\mathcal{D}_i = \{d_{i,1}, d_{i,2}, \cdots, d_{j,z_i}\}$ as the set of item reviews, where z_u and z_i denote the sizes of the sets $\mathcal{D}_u$ and $\mathcal{D}_i$, respectively. The model must make an accurate prediction $\hat{r}_{ui}$ using user-item pair (u, i) and its reviews $\mathcal{D}_u$ and $\mathcal{D}_i$.

3.2 Statically shared embedding network

SSEN uses global behavior data to learn homophily in a coarse-grained way and produces statically shared embedding of different levels for users and items. TSE uses three similarity measures to calculate the similarity matrix. These measures consider review, interaction, and rating features to calculate similarity, respectively. After getting the similarity matrix, we use K-means to cluster the matrix (each row represents a user) and then get the users' group id. The statically shared embeddings are the embedding vectors of group ids. We use $\mathbf{s}_u^{re}, \mathbf{s}_u^{in}, \mathbf{s}_u^{ra}$ to denote review-level, interaction-level, and rating-level shared embedding vectors, respectively. The symbols of items are similar to users. In the remainder of this section, we describe three similarity measures.

3.2.1 Review statically shared embedding . The review-based similarity measure includes two parts, a word frequency similarity(WF) and a sentiment similarity(SS). The word frequency similarity calculates the difference of each word frequency. We use $max(\gamma_{uw}, \gamma_{vw})$ operation to normalize the words' difference. The formula of the first expression is

$$sim_{uv}^{WF} = \frac{1}{|W|} * \sum_{w \in W} exp(-\frac{|\gamma_{uw} - \gamma_{vw}|}{max(\gamma_{uw}, \gamma_{vw})}), \tag{1}$$

where W denotes a dictionary used in our method, γ_{uw} denotes the frequency of word w in reviews of user u. To get the sentiment similarity, we use a pre-train model to predict the sentiment scores. Then, the sentiment similarity expression is

$$sim_{uv}^{SS} = \frac{1}{|\mathcal{I}_u \cap \mathcal{I}_v|} * \sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} exp(-\frac{|se_{ui} - se_{vi}|}{max(se_{ui}, se_{vi})}), \tag{2}$$

where se_{ui} denotes the sentiment score predicted by senta[38]. The final formula of the review-based user similarity is:

$$sim_{uv}^{re} = sim_{uv}^{WF} * sim_{uv}^{SS}. \tag{3}$$

3.2.2 Interaction statically shared embedding . To calculate the interaction similarity, we use a difference-based similarity measure (SMD) [1].

$$sim_{uv}^{in} = \frac{1 - \frac{f}{m} + \frac{n_{12}}{n_1 + n_2}}{2} \tag{4}$$

where $f = |\mathcal{I}_u \cup \mathcal{I}_v| - |\mathcal{I}_u \cap \mathcal{I}_v|, n_{12} = |\mathcal{I}_u \cap \mathcal{I}_v|, n_1 = |\mathcal{I}_u|, n_2 = |\mathcal{I}_v|$. $1 - \frac{f}{m}$ discovers the differences in user behaviors, and $\frac{n_{12}}{n_1 + n_2}$ emphasizes the importance of intersection. The two parts complement each other and get the exact similarity. We reduce full reliance on the co-click items through the SMD, making full use of interaction information that includes all clicked and non-clicked items.

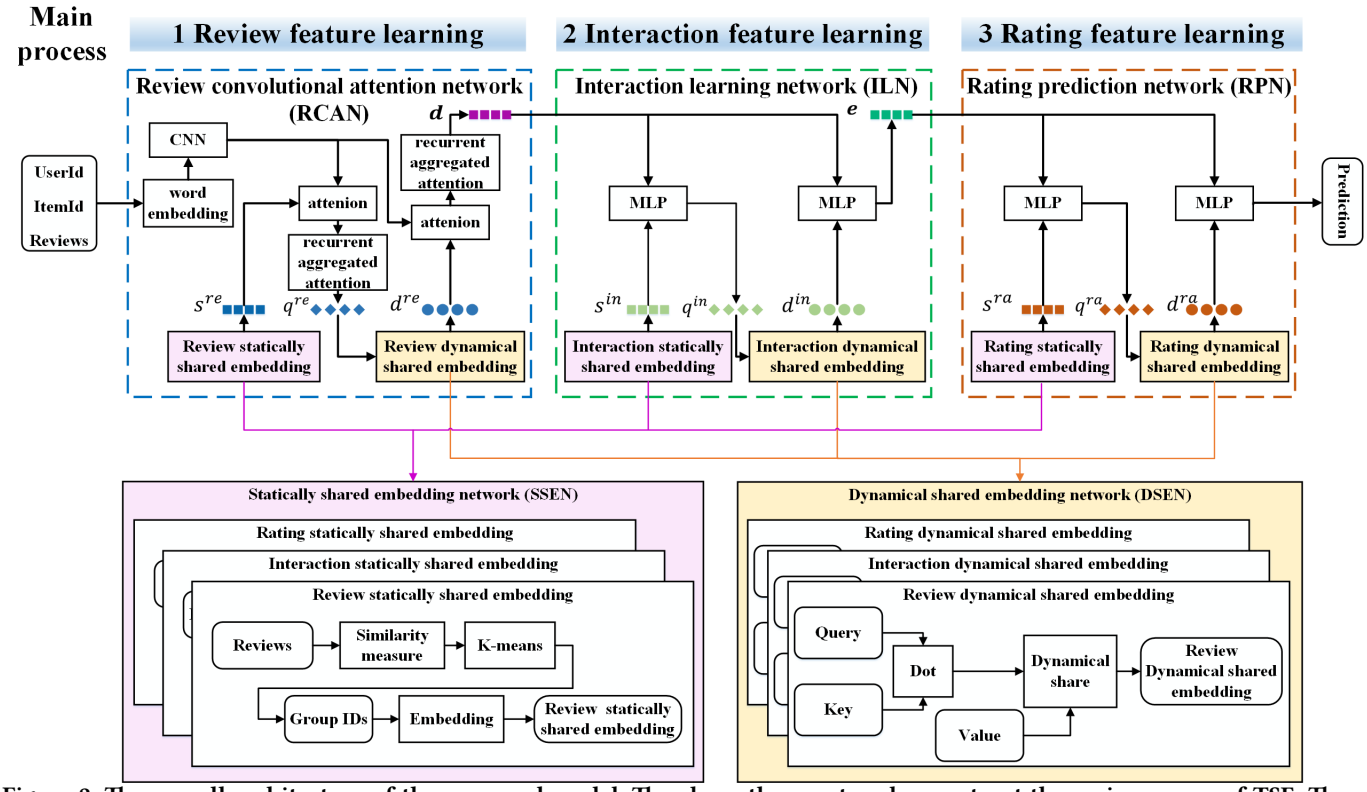


Figure 2: The overall architecture of the proposed model. The above three networks construct the main process of TSE. The statically shared embedding network and the dynamically shared embedding network are used to calculate embedding vectors from various features.

3.2.3 Rating statically shared embedding . The rating similarity measure has two parts[9]. TGhe first part considers the intersection of ratings, called the percentage of non common ratings (PNCR), and following:

$$sim_{uv}^{PNCR} = exp(-\frac{|\mathcal{I}| - |\mathcal{I}_u \cap \mathcal{I}_v|}{|\mathcal{I}|}). \tag{5}$$

$exp(-x)$ is a good kernel function for elementary similarity measure[9]. As a non-linear function, it can map the rating-based distance to a similarity score. There has the same trend of similarity and distance[10]. The second part calculates the absolute differences between the ratings (ADR), following:

$$sim_{uv}^{ADR} = \frac{\sum_{i \in \mathcal{I}} exp(-\frac{|r_{ui} - r_{vi}|}{max(r_{ui}, r_{vi})})}{|\mathcal{I}_u \cap \mathcal{I}_v|} \tag{6}$$

The final formula of rating-based user similarity is:

$$sim_{uv}^{ra} = sim_{uv}^{PNCR} * sim_{uv}^{ADR} \tag{7}$$

3.3 Dynamically shared embedding network

The dynamically shared embedding network models homophily in a fine-grained way by capturing and sharing dynamic preferences for each user-item pair. Figure 1 identifies that similar users have similar actions, and similar users could have similar dynamically high-level features in interactions. Thus, we share high-level features in the process of user-item interaction. $\mathbf{G}$ denotes a shared embedding table in a dynamically shared embedding layer. $\mathbf{e}_q$ denotes a high-level feature, which is used as the query to select their own dynamically shared embedding vector $\mathbf{e}_g$ from some presupposed feature vectors. Considering the diversity of user interests, we design two strategies for selection: a hard strategy that selects one out of the $|\mathbf{G}|$ groups, and a soft strategy that combines all embedding vectors with attention.

Hard strategy. The intuition behind our hard strategy is that only a single preference takes effect when a user makes a decision over items. It uses a query input vector to select the most similar shared embedding vectors. The final output is obtained as follows:

$$idx = \arg\max \ exp(\mathbf{q} * \mathbf{G}), \tag{8}$$

we select the idx-th row in $\mathbf{G}$ as the layer output.

$$\mathbf{e}_g = \mathbf{G}_{idx}. \tag{9}$$

Soft strategy. Sometimes, a user might like an item for various reasons, which have no distinct boundary. Instead of selecting the most prominent preference, the soft strategy is to combine multiple preferences via the attention mechanism:

$$\mathbf{e}_g = \sum_{g' \in G} softmax(\mathbf{q} * \mathbf{e}_{g'}) * \mathbf{e}_{g'}, \tag{10}$$

Independence regularization. Different from full embedding, shared embedding keeps a small embedding table, and each embedding vector represents a typical group. Embedding unique information will offer a useful dimension to characterize the behavioral patterns of groups. We use (11) as an independence regularization to constrain shared embedding, because (11) calculates mutual

information between embedding vectors and minimizing it constrains the similarity between embedding [42].

$$\mathcal{L}_{IND} = \sum_{\mathbf{e}_g \in \mathbf{G}} -log(\frac{exp(\mathbf{e}_g * \mathbf{e}_g)}{\sum\limits_{\mathbf{e}'_g \in \mathbf{G}} exp(\mathbf{e}_g * \mathbf{e}'_g)}) \tag{11}$$

where $\mathbf{G}$ is the embedding table, and $\mathbf{e}_g$ is one row of $\mathbf{G}$ and represent one embedding vector in $\mathbf{G}$.

3.4 Fusing two shared embedding networks in review-based recommendation

In TSE, RCAN, ILN, and RPN construct a general process. RCAN models the review features based on not only CNN but also review-level shared embedding; ILN models the interaction features based on not only review features but also interaction-level shared embedding; RPN makes predictions based on not only interaction features but also rating-level shared embedding. Each network in the process communicates with the two shared embedding networks. To fuse coarse- and fine-grained homophily, TSE adopts a whole-to-part principle. In each network, coarse-grained shared embedding and input data are used to calculate high-level features, which are used to select fine-grained shared embedding. The fine-grained shared embedding participates in the same process as the coarse-grained features to get the final output.

3.4.1 Review convolutional attention network. RCAN embeds words, produces feature vectors of reviews, and aggregates these vectors to represent users and items. First, we embed words of a review into a sequence of low-dimension dense embedding. $d = \{w_x\}_{x=1}^{p}$ denotes a review with p words. $\{\mathbf{w}_x\}_{x=1}^{p}$ is the word embedding sequence of d. Second, we apply CNN to capture the high-level word features $\mathbf{c}_x$. Third, an attention mechanism is used to summarize those word vectors with statically shared embedding vectors $\mathbf{s}_u^{re}$ and $\mathbf{s}_i^{re}$ into a review representation. Four, we use a recurrent aggregated attention layer to aggregate users' and items' reviews to represent themselves. Five, we input the representations into the review-level dynamically shared embedding layer to get the dynamically shared embedding $\mathbf{d}_u^{re}$ and $\mathbf{d}_i^{re}$. Six, $\mathbf{d}_u^{re}$, $\mathbf{d}_i^{re}$ and CNN-based features are input into attention layer. Seven, the result are input into the recurrent aggregated attention layer. We obtain the final review-based representation of users and items. We describe the details of the attention layer and the recurrent aggregated attention layer. Others are omitted for space reasons.

Attention. Different users may have different writing behaviors. Here we use the review-level shared embedding as a part of attention. The vector representation of a review d is as follows:

$$\mathbf{d} = \sum_{x=1}^{p} \alpha_x \mathbf{c}_x, \tag{12}$$

$$\alpha_x = \frac{exp(\omega^T \mathbf{c}_x)}{\sum_{x=1}^{p} exp(\omega^T \mathbf{c}_x)}, \tag{13}$$

$$\omega = \phi^{tanh}([\mathbf{s}_u^{re} : \mathbf{s}_i^{re}]), \tag{14}$$

where ϕ^{tanh} is a neural perceptron with $tanh$ as the activation function, and $\mathbf{s}_u^{re}$ and $\mathbf{s}_i^{re}$ are the review-level shared embedding.

Recurrent aggregated attention. Users could refer to other people's opinions when reviewing items [46]. Based on this, the recurrent aggregated attention layer models the correlation between users' and items' reviews using a mutual learning method. In the mutual learning method, textual semantic features are propagated to all users and items. As the output of the recurrent aggregated attention layer, the aggregated representation of users and items contains the wisdom of the public.

We first calculate the least Euclidean distance between $\{\mathbf{d}_{u,k}\}_{k=1}^{z_u}$ and $\{\mathbf{d}_{i,t}\}_{t=1}^{z_i}$ as attention scores to adjust the representation of the reviews, where $\{\mathbf{d}_{u,k}\}_{k=1}^{z_u}$ and $\{\mathbf{d}_{i,t}\}_{t=1}^{z_i}$ are the u's review vectors and i's review vectors, respectively.

$$\begin{gathered} e_{k,t}^{(\eta)} = |\mathbf{d}_{u,k}^{(\eta-1)} - \mathbf{d}_{i,t}^{(\eta-1)}|, \\ k = 1, ..., z_u; t = 1, ..., z_i; \eta = 1, ..., H \end{gathered} \tag{15}$$

where $e_{k,t}$ is the Euclidean distance between the k-th user review and the t-th item review. It determines the review features transferred between each user-item pair.

Then we aggregate review representations into a vector by the attention scores to represent the review information. We get the attention scores $\mathbf{a}^{(\eta)}$ and $\mathbf{b}^{(\eta)}$ from the negative of the minima distance. The initial representations $\mathbf{d}_{u,k}^{(0)}$ and $\mathbf{d}_{i,t}^{(0)}$ generated by the review convolutional attention network in parallel. The recurrent representation of the user side is as follows:

$$\delta_{u,k}^{(\eta)} = \frac{exp(a_k^{(\eta)})}{\sum_{k=1}^{z_u} exp(a_k^{(\eta)})}, \tag{16}$$

where $\delta_{u,k}^{\eta}$ is the η-th attention score of the review $d_{u,k}$.

$$\mathbf{t}_u^{(\eta)} = \sum_{k=1}^{z_u} \delta_{u,k}^{(\eta)} \mathbf{d}_{u,k}^{(\eta-1)}, \tag{17}$$

where $\mathbf{t}_u^{(\eta)}$ is the η-th aggregation of user u.

$$\mathbf{d}_{u,k}^{(\eta)} = \alpha \mathbf{d}_{u,k}^{(\eta-1)} + (1-\alpha)\mathbf{t}_u^{(\eta)}, \tag{18}$$

where $\mathbf{d}_{u,k}^{(\eta)}$ is the η-th representation of the review $d_{u,k}$.

$$\mathbf{h}_u^{(\eta)} = \alpha \mathbf{h}_u^{(\eta-1)} + (1-\alpha)\mathbf{t}_u^{(\eta)}. \tag{19}$$

where $\mathbf{h}_u^{(\eta)}$ is the η-th output of user u and it aggregates the output of the current layer and the previous $\eta-1$ layer. The item side calculation is similar to that of the user side. The initial representations $\mathbf{h}_u^1 = \mathbf{t}_u^{(1)}$ and $\mathbf{h}_i^1 = \mathbf{t}_i^{(1)}$. Through multiple iterative computations, this layer learns how reviews learn from each other. We use the final H-th output as the representations over the user and the item reviews, respectively.

$$\mathbf{d}_u = \mathbf{h}_u^{(H)},\ \mathbf{d}_i = \mathbf{h}_i^{(H)}. \tag{20}$$

3.4.2 Interaction learning network . Overfitting the review information is not always good for prediction [48]. As user click data are observational rather than experimental, bias is easily introduced into the data [7]. These biases cannot be learned from the characteristics of reviews. ILN tries to implicitly learn bias by using interaction-level shared embedding $\mathbf{s}_u^{in}$ and $\mathbf{s}_i^{in}$.

We first use MLP to aggregate review-based features and interaction-level shared embedding vectors to get the interaction feature.

$$\begin{aligned}\mathbf{e}' &= MLP_1(\mathbf{d}_u, \mathbf{s}_u^{in}, \mathbf{d}_i, \mathbf{s}_i^{in}) \\ &= \phi_x^{elu}(...\phi_2^{elu}(\phi_1^{elu}([\mathbf{d}_u; \mathbf{s}_u^{in} : \mathbf{d}_i : \mathbf{s}_i^{in}]))...),\end{aligned} \tag{21}$$

where ϕ_x^{elu} denotes the mapping function of x-th neural network layer. The interaction feature is used to select the interaction dynamically shared embedding $\mathbf{d}^{in}$. Finally, we use MLP to aggregate review-based features and $\mathbf{d}^{in}$ to get the final interaction feature.

$$\mathbf{e} = MLP_1(\mathbf{d}_u, \mathbf{d}_u^{in}, \mathbf{d}_i, \mathbf{d}_i^{in}). \tag{22}$$

3.4.3 Rating prediction network. Deep learning methods' superiority validates the advantage of nonlinearity, so we use MLP to make predictions [14]. We concatenate the interaction feature $\mathbf{e}$ and rating-level embedding vectors $\mathbf{s}_u^{ra}$ and $\mathbf{s}_i^{ra}$ to get a rating feature vector $\mathbf{e}_r$.

$$\mathbf{e}_r = MLP_1(\mathbf{e}, \mathbf{s}_u^{ra}, \mathbf{s}_i^{ra}). \tag{23}$$

Then $\mathbf{e}_r$ is input into the dynamically shared embedding network and gets the rating-level dynamically shared vector $\mathbf{d}^{ra}$. We combine the interaction feature and $\mathbf{d}^{ra}$ through an MLP to get the final prediction rating. The prediction function is as follows:

$$\begin{aligned}\hat{r}_{ui} &= MLP_2(\mathbf{e}_r, \mathbf{d}_u^{ra}, \mathbf{d}_i^{ra}) \\ &= \phi_{out}^{elu}(\phi_x^{elu}(...\phi_2^{elu}(\phi_1^{elu}([\mathbf{e} : \mathbf{s}_u^{ra} : \mathbf{s}_i^{ra}]))...)),\end{aligned} \tag{24}$$

where ϕ_{out}^{elu} notes the output layer.

3.5 Objective function

The objective function of TSE is defined as

$$\begin{aligned}\min \mathcal{L} = \sum_{(u,i,r_{ui}) \in dataset} (r_{ui} - \hat{r}_{ui})^2 + \lambda_F ||\Theta||_F + \\ \lambda_{IND}(\mathcal{L}_{IND}(\mathbf{G}^{re}) + \mathcal{L}_{IND}(\mathbf{G}^{in}) + \mathcal{L}_{IND}(\mathbf{G}^{ra})),\end{aligned} \tag{25}$$

where Θ is the model parameters, $||||_F$ is the Frobenius norm, $\mathcal{L}_{IND}$ is the independence regularization which we discussed in 3.3. λ_F and λ_{IND} are the regularized coefficient for the Frobenius norm and independence regularization, respectively. $\mathbf{G}^{re}$, $\mathbf{G}^{in}$, and $\mathbf{G}^{ra}$ are the embedding table in the review-, interaction-, and rating-level dynamical shared embedding layers, respectively. We use an Adam optimizer to optima our objective function.

4 EXPERIMENTS

4.1 Experimental setup

4.1.1 Datasets. We choose five datasets that came from Amazon[1] to validate the performance of ADSE, i.e., Sports and Outdoors(SO), Video Games(VG), Grocery and Gourmet Food(GGF), Office Products(OP), Music Instruments(MI). All words in reviews are processed into the lower cases, and the stop words are filtered [45]. A vocabulary includes 20,000 common words that come from datasets. The review length was standardized to cover 90% of reviews. Reviews in the test are not available [4]. Table 1 shows the statistics of the datasets.

[1]http://jmcauley.ucsd.edu/data/amazon/

Table 1: Statistics of the datasets.

Dataset	#user	#item	#rating	Density
MI	1,429	900	10,261	0.798%
OP	4905	2420	53,228	0.449%
GGF	14,681	8,713	151,254	0.118%
VG	24,303	10,672	231,780	0.089%
SO	35,598	18,357	296,337	0.045%

4.1.2 Baselines. To demonstrate the advantage of the proposed model, we compare it with the following models, PMF [33], GNMF [3], SVR-TC [30], SVD-MCOC [2], KL-KM [8], NeuMF [14], CDL [41], ConvMF [19], DeepCoNN [50], D-attn [34], NARRE [5], CARL [45], DAML [24], HTI [44],TAERT [13], and DSRLN [27].

4.1.3 Experimental setup. We use GloVe-100 [32] to initialize the word embedding. The kernel sizes of two CNN are 3 and 5, which is one of the best choices in existing methods [5, 19, 24]. The neural network has 3 hidden layers and the output dimension of these layers are 300, 200, and 100. The dropout rate is 0.5, the batch size is 128, and the initial learning rate of the Adam optimizer is 0.0001. In TSE, each user has three embedding vectors, but we calculate interaction-based similarities based on ratings, which leads it to be similar to rating-based similarities. Thus, we set the K in each shared embedding layer to the square root of its scale of the input, making the representation of each user to keep personalized.

We choose two types of metrics to evaluate the model performance, including two error-based metrics and a ranking-based metric. The error-based metrics are Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), which are used in most related works [13, 24, 33]. Each dataset is randomly split into three parts, i.e., training set, validation set (10%), and test set (10%). The ranking-based metric is normalized discounted cumulative gain NDCG@K ($K = 10$ in this paper). For each user, 80% of the ratings are randomly selected for training, and the remaining 20% are selected for testing. All experiment results are the mean of 5 times.

4.2 Experiment result

4.2.1 Error-based metric. To identify the outstanding aspect of TSE, we give a full performance comparison with all baselines. The overall performances of all baseline methods and TSE are reported in Table 2. The best results and the best baseline are highlighted in boldface and underlined, respectively.

TSE outperforms the baselines on the five datasets. TSE (H) denotes the TSE with hard strategy and TSE (S) denotes TSE with soft strategy. On RMSE, TSE(H) outperforms the best baseline by 16.79% (MI), 18.86% (OP), 10.87% (GGF), 28.09% (VG), and 18.50% (SO). TSE (H) achieves an average improvement of 18.62% in RMSE and 22.99% in MAE over the TSE (S) outperforms the best baseline by 20.50% in RMSE and 23.96% in MAE on average. The result implies that TSE effectively predicts ratings on datasets. Existing methods learn one or more embedding tables for users and items. The high sparsity of the recommendation dataset makes them difficult to converge to the optimal result. Through TSE, embedding vectors are fully trained sufficiently and achieve good performance. The fusion of coarse- and fine-grained homophily has significantly improved the quality of recommended service.

Except in MI, TSE(S) is better than TSE(H). The five datasets represent five different fields. In MI, people usually focus on the price, quality, and sound of musical instruments, which leads to

Table 2: Performance comparison across the dataset for all models. The best results and the best baseline are highlighted in boldface and underlined respectively. We also compare the best results with the best baseline and report the improvement (Best Imp.).

Dataset	MI		OP		GGF		VG		SO	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
PMF	1.352	1.137	1.43	1.265	1.572	1.397	1.606	1.395	1.376	1.203
GNMF	0.921	0.702	0.932	0.745	1.086	0.871	1.203	0.946	0.990	0.772
SVR-TC	1.051	0.599	1.031	0.662	1.138	0.708	1.269	0.877	1.087	0.637
SVD-MCOC	0.955	0.715	0.917	0.710	1.041	0.789	1.163	0.933	0.955	0.737
KL-KM	1.024	0.672	0.936	0.671	1.077	0.775	1.149	0.848	1.004	0.696
NeuMF	0.904	0.720	0.921	0.730	1.197	0.943	1.103	0.870	0.979	0.752
CDL	1.080	0.834	1.223	1.062	1.190	0.967	1.165	0.902	1.089	0.852
ConvMF	1.026	0.786	0.952	0.728	1.092	0.863	1.145	0.899	1.013	0.824
DeepCoNN	1.003	0.759	0.901	0.711	1.036	0.802	1.168	0.875	0.885	0.719
D-attn	0.956	0.742	0.923	0.716	1.070	0.824	1.062	0.842	0.997	0.784
NARRE	0.922	0.695	0.867	0.681	0.963	0.747	1.039	0.799	0.882	0.690
CARL	0.878	0.677	0.834	0.647	0.961	0.753	1.029	0.798	0.888	0.686
DAML	0.848	0.651	0.811	0.612	0.938	0.735	1.045	0.788	0.883	0.667
HTI	0.813	0.611	0.731	0.552	0.877	0.672	0.966	0.731	0.826	0.629
TAERT	0.793	0.592	0.829	0.561	0.816	0.646	0.921	0.647	0.807	0.599
DSRLN	1.036	0.620	0.890	0.621	0.952	0.721	0.983	0.754	0.900	0.651
TSE(H)	**0.679**	**0.506**	0.615	0.477	0.736	0.523	0.719	0.524	**0.681**	0.443
TSE(S)	0.723	0.511	**0.568**	**0.464**	**0.697**	**0.520**	**0.707**	**0.518**	0.691	**0.441**
Best Imp.(%)	16.79	17.00	28.70	18.97	17.07	24.23	30.27	24.90	18.50	35.83

Table 3: NDCG@10 performance for all models.

	MI	OP	GGF	VG	SO
DeepCoNN	0.977	0.973	0.969	0.971	0.975
NARRE	0.978	0.976	0.967	0.968	0.977
CARL	0.980	0.978	0.969	0.969	0.977
DAML	0.982	0.978	0.972	0.974	0.981
HTI	0.982	0.980	0.973	0.976	0.981
TAERT	0.980	**0.981**	0.967	**0.979**	0.975
DSRLN	0.978	0.974	0.967	0.971	0.977
TSE(H)	**0.985**	**0.981**	**0.979**	**0.979**	**0.986**
TSE(S)	0.983	**0.981**	0.978	**0.979**	**0.986**

TSE(H) having the best performance. In other files, users are usually willing to try some other types of items in addition to their main interests. This is the reason why TSE(S) performs well.

4.2.2 Ranking-based metric. We further compare TSE with the latest deep learning-based models, including DeepCoNN, NARRE, CARL, DAML, HTI, TAERT, and DSRLN. The NDCG@10 results are shown in Table 3. TSE outperforms the baselines on the five datasets. NDCG is less sensitive to the accuracy of the prediction results. Even if the prediction error is large, as long as the ranking results are not affected, NDCG remains unchanged. The impact of data sparsity leads to fewer test samples, which also leads to overall higher results. TSE significantly outperforms the baseline on the rating prediction task, which is very important for recommender systems. In practical application scenarios, the revenue of recommending an item is unknown. Therefore, accurate rating prediction is very important for estimating the revenue of recommendation results.

4.3 Model complexity

A major problem with deep learning is model complexity which increases with performance improvement. Some experimental results show a positive correlation between models' performance and complexity [44]. Therefore, we compare TSE with the latest deep learning-based models which have at least one embedding layer and study the model complexity of these models. The compared models include DeepCoNN, NARRE, CARL, DAML, HTI, TAERT, RGNN, and DSRLN. All models were trained using an i7-11700K CPU with 128 GB RAM and an RTX-3090 GPU with 24 GB RAM. The results are shown in Table 4. TSE(H) and TSE(S) have the same network structures so they have the same parameters.

From Table 4, we can see that TSE has the smallest number of parameters on four datasets. The number of parameters in most models is increased with the size of the dataset. The parameters of a model can be divided into two parts, prediction parameters, and embedding tables. The increment due to the scale of users and items makes a large embedding table. The number of parameters in TSE is stable as it uses shared embedding to replace full embedding. The number of parameters of shared embedding is decided by the number of groups, not the scale of users and items. This is the reason why the number of parameters in TSE has a small change with the increasing dataset scale. In conclusion, TSE has a good performance of rating predictions with small model complexity.

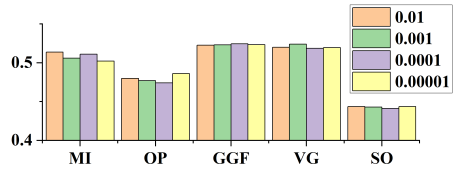
4.4 Analysis

4.4.1 Impact of the independence regularization. To demonstrate the sensitivity of independence regularization, we experimentally investigate the effect of regularization. The regularization of shared embedding is important for the performance of the recommendation task. Therefore, we study the effect of regularization by varying its value. We explored the coefficient of the regularization λ_{IND} from $\{0.01, 0.001, 0.0001, 0.00001\}$. The regularization effect on the model's performance is shown in Figure.3.

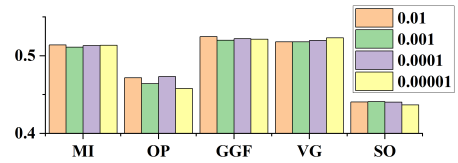
The hard strategy is sensitive to changes in the regularization coefficient. Moreover, TSE(H) achieves the best performance at 0.001

Table 4: Trainable parameters of various models. We merge the results of TSE(H) and TSE(S) as they have the same parameter. The best and the second-best results are highlighted by boldface and underlined respectively.

	MI	OP	GGF	VG	SO
DeepCoNN	**2.42M**	2.92M	4.53M	5.69M	7.59M
NARRE	3.19M	7.89M	11.8M	29.4M	19.6M
CARL	3.15M	3.23M	3.49M	3.67M	6.98M
DAML	2.45M	2.95M	4.56M	5.72M	7.62M
HTI	2.45M	2.95M	4.56M	5.71M	7.61M
TAERT	3.07M	3.58M	5.22M	6.40M	8.33M
DSRLN	2.55M	3.55M	6.76M	9.08M	12.9M
TSE	2.63M	**2.66M**	**2.71M**	**2.74M**	**2.78M**



(a) MAE on hard strategy



(b) MAE on soft strategy

Figure 3: Impact of the independence regularization.

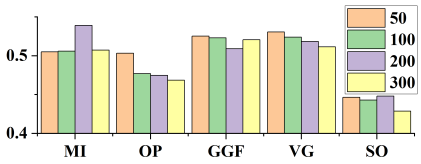
in most datasets. For example, the style differences between musical instruments are significant. The reasonable use of independent regularization can enable the model to learn the differences between different musical instruments. The results of TSE(S) are less differentiated because the soft strategy mixes different feature vectors. In conclusion, the model performs best when the regularization coefficient is fixed at 0.001.

4.4.2 Impact of the embedding dimension. Since embedding users or items contains rich user preferences and product feature information, the dimension of embedding vectors is essential for representing users and items. We explore the impact of the embedding dimension. Because we use GloVe to embed all words, we experiment to identify the effect of four dimensions, {50, 100, 200, 300}, used in GloVe. The experiment results are reported in Figure 4.

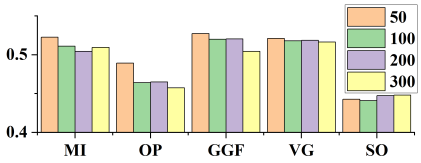
By comparing the performance difference between the four dimensions, we find that the effect of embedding dimension on performance are 5.77% (TSE(H)) and 5.63%(TSE(S)) on average. In the meantime, as the embedding dimension increases, the model performance first increases and then decreases in some datasets.

In summary, TSE will achieve the best performance in most datasets when its embedding dimension is 100. Big embedding dimensions have great performance but with big model complexity.

4.4.3 Impact of the statically shared embedding network. To demonstrate the effect of the statically shared embedding network in TSE, we design two variants and do a comparison experiment. The first variant is a global adaptive model (GA). GA keeps a full embedding for users and items as the query input for shared vectors. The



(a) MAE on hard strategy



(b) MAE on soft strategy

Figure 4: Experiment results on embedding dimensions.

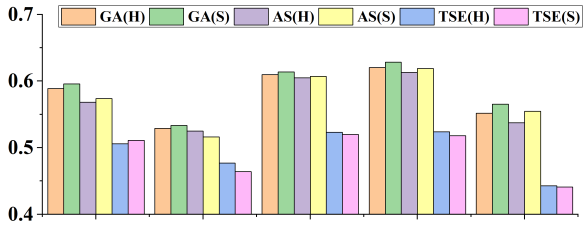
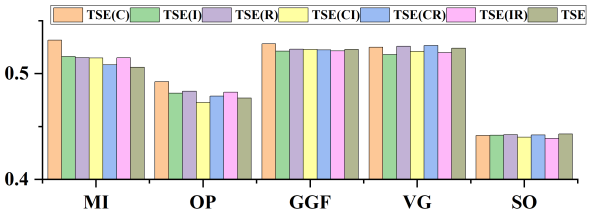


Figure 5: The MAE performance of TSE and the two variants is compared to verify the impact of the statically shared embedding network.

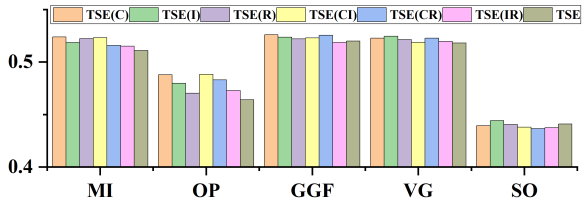
computational flow of this method is similar to that we use full embedding instead of the statically shared embedding network. The second variant, an adaptive signal model (AS), predicts without the statically shared embedding network. In TSE, the output of the statically shared embedding network is used to calculate the query vectors. However, in AS, we calculate the query vectors by using review features.

Figure 5 illustrates the experiment results. Overall, TSE achieves the best results on the five datasets. TSE performs better than GA and SA showing that the statically shared embedding network uses extra knowledge to give pre-groups, helping TSE achieve better performance than without it. Through shared embedding, users are also allowed to transfer knowledge and benefit from their own group, which improves recommendation performances. The model's learning of one user can benefit from the behavior of similar users, as the sparse history data of one user is not enough to train that person's preference vector with full embedding. Shared embedding aggregates the data of several users to train one vector, providing more opportunities to realize better performance. AS is better than GA indicating that insufficiently trained fully embedding vectors are harmful to model performance. In conclusion, TSE improves prediction results by combining extra knowledge and model inputs to adjust shared vectors based on various information.

4.4.4 Impact of the dynamically shared embedding network. Dynamically shared embedding network models homophily in a fine-grained way. To identify its effect, we first compare TSE with a variant that is without the dynamically shared embedding network. We use TSE(-D) to denote it. The experiment results are shown



(a) RMSE on soft strategy



(b) MAE on soft strategy

Figure 6: Impact of various dynamically shared embedding layers.

in Table 5, which illustrates the effectiveness of the dynamically shared embedding network.

To further get deep insights into the dynamically shared embedding network and its effect based on three features, we do an ablation study to investigate its impact on different features, i.e., three shared embedding layers. For the sake of simplicity, we write TSE(C) as the layer only used to adjust the group result based on reviews, adaptively TSE(I) is the layer only used for interaction, and TSE(R) is the layer only used for rating. Also, we study some combinations of these layers, i.e., TSE(CI), TSE(CR), TSE(IR). Figure 6 plots the performance comparison of those methods.

Figure 6 illustrates the influence of different layers on the final result. The corresponding features can be learned using a single dynamic layer. Moreover, the order of performance of these three methods is TSE(C) < TSE(I) < TSE(R). The reason is that the search space for features is different. The text feature space is larger than the interaction space between users and items. The interaction space is larger than the rating space. Sometimes, review and rating information have a degree of discrepancy. The results are usually not as good as using either of the two alone when directly fusing the group result based on the review feature and rating feature together. TSE introduces and successfully fuses three dynamically shared embedding layers, achieving the best performance. Overall, TSE performs better than its variant, identifying that through the dynamically shared embedding network successfully models homophily in a fine-grained way.

4.4.5 Case study for rating distribution. Data sparsity manifests itself as items, where a small number of items occupy many ratings, and the rest have very few ratings. These items with few ratings are referred to as long-tail items. To further validate the effectiveness of TSE, we analyzed the prediction results of TSE on different rating distributions. We chose VG dataset to do this experiment as TSE achieved better improvements on RMSE than other datasets. The distribution of VG is long-tailed. The top 6.11% of items hold 34.76% ratings.

Table 5: Comparison results of TSE and a variant that is without the dynamically shared embedding network.

	MI	OP	GGF	VG	SO
TSE(-D)	0.518	0.499	0.534	0.540	0.462
TSE(H)	0.506	0.477	0.523	0.524	0.443
TSE(S)	0.511	0.464	0.520	0.518	0.441

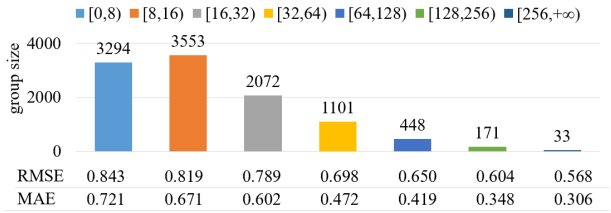


Figure 7: Model performance on different groups after grouping by rating distribution.

In this experiment, we first counted the number of ratings for each item and grouped the items according to the results by Log Base 2. Then we counted the proportion of the total number of ratings in each grouping to the total ratings and the prediction performance of the model in that grouping. The experiment results are shown in Figure 7. Their RMSE and MAE performances are better than average in the whole dataset (0.707 on RMSE and 0.518 on MAE). When items have at least 32 ratings, they will achieve average performance. Compared with TAERT achieving 0.921 on RMSE, TSE significantly improves the performance of long-tail items. By sharing embedding vectors, the prediction performance of TSE for long-tail items is improved with the help of head items.

5 CONCLUSION

In this work, we propose a novel review-based recommendation model, namely two-tier shared embedding (TSE), which incorporates the advantages of coarse- and fine-grained ways of modeling homophily. TSE considers global behaviors to model homophily in a coarse-grained way, and the high-level feature in the process of each user-item interaction to model homophily in a fine-grained way. In the review-based recommendation, TSE fuses these two ways, making accurate predictions. The experimental results on five real-world datasets show that TSE consistently outperforms state-of-the-art review-based recommendation methods in terms of RMSE and MSE. In future work, we plan to investigate graph-based clustering algorithms to propose a more lightweight and flexible model. In addition, group recommendation methods give some insight that can be used to improve our solution further.

ACKNOWLEDGMENTS

This work is partially supported by the National Key R&D Program of China (No.2022YFB3103100), the Major Research Plan of the National Natural Science Foundation of China (No.92167102), the Importation and Development of High-Caliber Talents Project of Beijing Municipal Institutions (CIT&TCD20190308), the Project of Beijing Municipal Education Commission (No.KM202110005025), Beijing Municipal Natural Science Foundation Project (No.Z200002), and Engineering Research Center of Intelligent Perception and Autonomous Control, Ministry of Education.

REFERENCES

[1] Ali A. Amer, Hassan Ismail Abdalla, and Loc Nguyen. 2021. Enhancing recommendation systems performance using highly-effective similarity measures. *Knowl. Based Syst.* 217 (2021), 106842. https://doi.org/10.1016/j.knosys.2021.106842

[2] Jiajun Bu, Xin Shen, Bin Xu, Chun Chen, Xiaofei He, and Deng Cai. 2016. Improving Collaborative Recommendation via User-Item Subgroups. *IEEE Trans. Knowl. Data Eng.* 28, 9 (2016), 2363–2375. https://doi.org/10.1109/TKDE.2016.2566622

[3] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S. Huang. 2011. Graph Regularized Nonnegative Matrix Factorization for Data Representation. *IEEE Trans. Pattern Anal. Mach. Intell.* 33, 8 (2011), 1548–1560. https://doi.org/10.1109/TPAMI.2010.231

[4] Rose Catherine and William W. Cohen. 2017. TransNets: Learning to Transform for Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*, Paolo Cremonesi, Francesco Ricci, Shlomo Berkovsky, and Alexander Tuzhilin (Eds.). ACM, 288–296. https://doi.org/10.1145/3109859.3109878

[5] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference*. 1583–1592.

[6] Hongqi Chen, Zhiyong Feng, Shizhan Chen, Xiao Xue, Hongyue Wu, Yingchao Sun, Yanwei Xu, and Gaoyong Han. 2022. Capturing Users' Fresh Interests via Evolving Session-Based Social Recommendation. In *IEEE International Conference on Web Services, ICWS 2022, Barcelona, Spain, July 10-16, 2022*, Claudio Agostino Ardagna, Nimanthi L. Atukorala, Boualem Benatallah, Athman Bouguettaya, Fabio Casati, Carl K. Chang, Rong N. Chang, Ernesto Damiani, Chirine Ghedira Guegan, Robert Ward, Fatos Xhafa, Xiaofei Xu, and Jia Zhang (Eds.). IEEE, 182–187. https://doi.org/10.1109/ICWS55610.2022.00039

[7] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and debias in recommender system: A survey and future directions. *arXiv preprint arXiv:2010.03240* (2020).

[8] Jiangzhou Deng, Junpeng Guo, and Yong Wang. 2019. A Novel K-medoids clustering recommendation algorithm based on probability distribution for collaborative filtering. *Knowl. Based Syst.* 175 (2019), 96–106. https://doi.org/10.1016/j.knosys.2019.03.009

[9] Achraf Gazdar and Lotfi Hidri. 2020. A new similarity measure for collaborative filtering based recommender systems. *Knowl. Based Syst.* 188 (2020). https://doi.org/10.1016/j.knosys.2019.105058

[10] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2013. A Novel Bayesian Similarity Measure for Recommender Systems. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, Francesca Rossi (Ed.). IJCAI/AAAI, 2619–2625. http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6615

[11] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2015. Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems. *Knowl. Based Syst.* 74 (2015), 14–27. https://doi.org/10.1016/j.knosys.2014.10.016

[12] Lei Guo, Hongzhi Yin, Tong Chen, Xiangliang Zhang, and Kai Zheng. 2022. Hierarchical Hyperedge Embedding-Based Representation Learning for Group Recommendation. *ACM Trans. Inf. Syst.* 40, 1 (2022), 3:1–3:27. https://doi.org/10.1145/3457949

[13] Siyuan Guo, Ying Wang, Hao Yuan, Zeyu Huang, Jianwei Chen, and Xin Wang. 2021. TAERT: Triple-Attentional Explainable Recommendation with Temporal Convolutional Network. *Inf. Sci.* 567 (2021), 185–200. https://doi.org/10.1016/j.ins.2021.03.034

[14] 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.

[15] Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Wei Cao. 2014. Deep Modeling of Group Preferences for Group-Based Recommendation. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, Carla E. Brodley and Peter Stone (Eds.). AAAI Press, 1861–1867. http://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8458

[16] Chunli Huang, Wenjun Jiang, Jie Wu, and Guojun Wang. 2020. Personalized Review Recommendation based on Users' Aspect Sentiment. *ACM Trans. Internet Techn.* 20, 4 (2020), 42:1–42:26. https://doi.org/10.1145/3414841

[17] Wang-Cheng Kang, Derek Zhiyuan Cheng, Tiansheng Yao, Xinyang Yi, Ting Chen, Lichan Hong, and Ed H. Chi. 2021. Learning to Embed Categorical Features without Embedding Tables for Recommendation. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, Feida Zhu, Beng Chin Ooi, and Chunyan Miao (Eds.). ACM, 840–850. https://doi.org/10.1145/3447548.3467304

[18] Ondrej Kassák, Michal Kompan, and Mária Bieliková. 2016. Personalized hybrid recommendation for group of users: Top-N multimedia recommender. *Inf. Process. Manag.* 52, 3 (2016), 459–477. https://doi.org/10.1016/j.ipm.2015.10.001

[19] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM conference on recommender systems*. 233–240.

[20] Chin-Hui Lai and Chia-Yu Hsu. 2021. Rating prediction based on combination of review mining and user preference analysis. *Information Systems* 99 (2021), 101742.

[21] 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, CIKM 2019, Beijing, China, November 3-7, 2019*, Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.). ACM, 2615–2623. https://doi.org/10.1145/3357384.3357814

[22] Chenliang Li, Cong Quan, Li Peng, Yunwei Qi, Yuming Deng, and Libing Wu. 2019. A Capsule Network for Recommendation and Explaining What You Like and Dislike. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer (Eds.). ACM, 275–284. https://doi.org/10.1145/3331184.3331216

[23] Man Li, Luosheng Wen, and Feiyu Chen. 2021. A novel Collaborative Filtering recommendation approach based on Soft Co-Clustering. *Physica A: Statistical Mechanics and its Applications* 561 (2021), 125140.

[24] Donghua Liu, Jing Li, Bo Du, Jun Chang, and Rong Gao. 2019. Daml: Dual attention mutual learning between ratings and reviews for item recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 344–352.

[25] Hongtao Liu, Wenjun Wang, Qiyao Peng, Nannan Wu, Fangzhao Wu, and Pengfei Jiao. 2021. Toward Comprehensive User and Item Representations via Three-tier Attention Network. *ACM Trans. Inf. Syst.* 39, 3 (2021), 25:1–25:22. https://doi.org/10.1145/3446341

[26] Hongtao Liu, Wenjun Wang, Hongyan Xu, Qiyao Peng, and Pengfei Jiao. 2020. Neural Unified Review Recommendation with Cross Attention. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 1789–1792. https://doi.org/10.1145/3397271.3401249

[27] Tongcun Liu, Siyuan Lou, Jianxin Liao, and Hailin Feng. 2022. Dynamic and Static Representation Learning Network for Recommendation. *IEEE Transactions on Neural Networks and Learning Systems* (2022), 1–11. https://doi.org/10.1109/TNNLS.2022.3177611

[28] Yong Liu, Susen Yang, Yinan Zhang, Chunyan Miao, Zaiqing Nie, and Juyong Zhang. 2021. Learning hierarchical review graph representations for recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[29] Giseop Noh, Hayoung Oh, and Jaehoon Lee. 2018. Power users are not always powerful: The effect of social trust clusters in recommender systems. *Inf. Sci.* 462 (2018), 1–15. https://doi.org/10.1016/j.ins.2018.05.058

[30] Yoon-Joo Park. 2013. The Adaptive Clustering Method for the Long Tail Problem of Recommender Systems. *IEEE Trans. Knowl. Data Eng.* 25, 8 (2013), 1904–1915. https://doi.org/10.1109/TKDE.2012.119

[31] Yoon-Joo Park and Alexander Tuzhilin. 2008. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*, Pearl Pu, Derek G. Bridge, Bamshad Mobasher, and Francesco Ricci (Eds.). ACM, 11–18. https://doi.org/10.1145/1454008.1454012

[32] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[33] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis (Eds.). Curran Associates, Inc., 1257–1264. https://proceedings.neurips.cc/paper/2007/hash/d7322ed717dedf1eb4e6e52a37ea7bcd-Abstract.html

[34] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the eleventh ACM conference on recommender systems*. 297–305.

[35] Leily Sheugh and Sasan Hossein Alizadeh. 2018. A novel 2D-Graph clustering method based on trust and similarity measures to enhance accuracy and coverage in recommender systems. *Inf. Sci.* 432 (2018), 210–230. https://doi.org/10.1016/j.ins.2017.12.007

[36] Liye Shi, Wen Wu, Wang Guo, Wenxin Hu, Jiayi Chen, Wei Zheng, and Liang He. 2022. SENGR: Sentiment-Enhanced Neural Graph Recommender. *Inf. Sci.* 589 (2022), 655–669. https://doi.org/10.1016/j.ins.2021.12.120

[37] Jiliang Tang, Huiji Gao, Xia Hu, and Huan Liu. 2013. Exploiting homophily effect for trust prediction. In *Sixth ACM International Conference on Web Search and Data Mining, WSDM 2013, Rome, Italy, February 4-8, 2013*, Stefano Leonardi, Alessandro Panconesi, Paolo Ferragina, and Aristides Gionis (Eds.). ACM, 53–62.

https://doi.org/10.1145/2433396.2433405

[38] Hao Tian, Can Gao, Xinyan Xiao, Hao Liu, Bolei He, Hua Wu, Haifeng Wang, and Feng Wu. 2020. SKEP: Sentiment Knowledge Enhanced Pre-training for Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 4067–4076. https://doi.org/10.18653/v1/2020.acl-main.374

[39] Lyle H Ungar and Dean P Foster. 1998. Clustering methods for collaborative filtering. In *AAAI workshop on recommendation systems*, Vol. 1. Menlo Park, CA, 114–129.

[40] Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 448–456.

[41] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1235–1244.

[42] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. 2021. Learning Intents behind Interactions with Knowledge Graph for Recommendation. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 878–887. https://doi.org/10.1145/3442381.3450133

[43] Zhen Wang, Chang-Ai Sun, and Marco Aiello. 2022. Context-aware IoT Service Recommendation: A Deep Collaborative Filtering-based Approach. In *IEEE International Conference on Web Services, ICWS 2022, Barcelona, Spain, July 10-16, 2022*, Claudio Agostino Ardagna, Nimanthi L. Atukorala, Boualem Benatallah, Athman Bouguettaya, Fabio Casati, Carl K. Chang, Rong N. Chang, Ernesto Damiani, Chirine Ghedira Guegan, Robert Ward, Fatos Xhafa, Xiaofei Xu, and Jia Zhang (Eds.). IEEE, 150–159. https://doi.org/10.1109/ICWS55610.2022.00035

[44] Jiahui Wen, Jingwei Ma, Hongkui Tu, Mingyang Zhong, Guangda Zhang, Wei Yin, and Jian Fang. 2020. Hierarchical text interaction for rating prediction. *Knowledge-Based Systems* 206 (2020), 106344.

[45] Libing Wu, Cong Quan, Chenliang Li, Qian Wang, Bolong Zheng, and Xiangyang Luo. 2019. A context-aware user-item representation learning for item recommendation. *ACM Transactions on Information Systems (TOIS)* 37, 2 (2019), 1–29.

[46] Shiwen Wu, Yuanxing Zhang, Wentao Zhang, Kaigui Bian, and Bin Cui. 2021. Enhanced review-based rating prediction by exploiting aside information and user influence. *Knowledge-Based Systems* (2021), 107015.

[47] Bin Xu, Jiajun Bu, Chun Chen, and Deng Cai. 2012. An exploration of improving collaborative recommender systems via user-item subgroups. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, Alain Mille, Fabien Gandon, Jacques Misselis, Michael Rabinovich, and Steffen Staab (Eds.). ACM, 21–30. https://doi.org/10.1145/2187836.2187840

[48] Yuanbo Xu, Yongjian Yang, Jiayu Han, En Wang, Fuzhen Zhuang, Jingyuan Yang, and Hui Xiong. 2019. NeuO: Exploiting the sentimental bias between ratings and reviews with neural networks. *Neural Networks* 111 (2019), 77–88.

[49] Hongzhi Yin, Qinyong Wang, Kai Zheng, Zhixu Li, and Xiaofang Zhou. 2022. Overcoming Data Sparsity in Group Recommendation. *IEEE Trans. Knowl. Data Eng.* 34, 7 (2022), 3447–3460. https://doi.org/10.1109/TKDE.2020.3023787

[50] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the tenth ACM international conference on web search and data mining*. 425–434.