Received March 9, 2020, accepted March 17, 2020, date of publication March 19, 2020, date of current version March 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2982037

RACRec: Review Aware Cross-Domain Recommendation for Fully-Cold-Start User

YARU JIN, SHOUBIN DONG, YONG CAI, AND JINLONG HU
School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

Corresponding author: Shoubin Dong (sbdong@scut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61976239, in part by the Guangzhou Science and Technology Program under Grant 201903010046, and in part by the Zhongshan Innovation Foundation of High-End Scientific Research Institutions under Grant 2019AG031.

ABSTRACT Traditional recommendation algorithms such as matrix factorization, collaborative filtering perform poorly when lack of interactive information of user and product, known as the user cold-start problem, which may cut down the revenue of E-Commerce platform. Moreover, it is more challenging to generate recommendation lists for users who have no information at all because there is no preference information about them that could be leveraged, which is the user fully-cold-start problem. In this paper, a review aware cross-domain recommendation algorithm, called RACRec, is proposed to address the fully-cold-start problem in the field of product recommendation. Firstly, reviews are dynamically selected by using the adjacency matrix. Secondly, domain-specific preference vectors and domain-shared preference vectors of the cold start user are extracted by a migration model. On the other hand, the product feature vector in the target domain, which is generated from review texts by encoder and decoder, is combined with preference vectors of the cold-start user to make the rating prediction. Experiments on the Amazon datasets reveal that RACRec outperforms the state-of-the-art recommendation algorithms.

INDEX TERMS Cross-domain recommendation, select reviews, fully-cold-start, review aware recommendation.

I. INTRODUCTION

Nowadays, people are more willing to shopping online. Facing the various choice of products provided by E-platforms, people always have difficulty picking up the interesting one. Recommender System [1] is considered an effective way to solve such a problem, which led to the development of traditional recommendation algorithms, such as Matrix Factorization [2]–[4], Collaborative Filtering [5] and so on. In order to meet the increasingly diverse user demand of E-commerce, platforms usually divide their products into many domains, such as Books, Movies, CDs, just like what Amazon has done, which is a convenience for the planner of the platform to hold activities to attract users to spend in the target domain. Among those activities, attracting users who have never shopped in the target domain to buy satisfactory products are the most crucial goals. However, such new users have no interactive information, so that the above methods suffer from User Cold Start Problem [6], especially in the field of E-Commerce platform. In such a situation, traditional recommendation algorithms often unable to generate valid product recommendation lists for cold-start users.

The user profiling model is an important part of a product recommender system, for it provides the preference information of a user, which could be used to pick the interesting products up by the product recommender system. So, in the modern E-Commerce platform, such as Amazon, Tmall, and JingDong, there are several feedback ways provided to make opinions express more conveniently. Among these feedback ways, the overall rating is a high-level one. Users usually give a rating for a product after considering multiple aspects of it. For example, when a user is giving a rating for a phone, what he will consider are the size of the screen and the price of the phone, etc. But what can we learn from the overall rating is just the high-level information, not his opinion about the screen and the price. On the other hand, review text is a common way that E-Commerce platforms provide users to express their detailed opinions about the product. For example, one may give a review that "I like the big screen of that phone, and the price is nice." What the recommender system can learn from the review is that the user likes the big screen, and the value of money is something he may seriously consider when buying a phone. So, the

The associate editor coordinating the review of this manuscript and approving it for publication was Guanjun Liu.

review text feedback is a way to compensate for the overall rating feedback, especially when there are very few purchase records. There is no doubt that the recommender system could model a more precise user preference model by mining such information from review texts.

It is well known that news has headlines, videos have titles, and products have attribute characteristics and short descriptions, which may help generate feature vector representation. However, a product has dozens of attributes. The importance of these attribute characteristics is quite different. For example, the camera pixel weight of smartphones with high-quality cameras as selling points should be more significant. For music phones, the category of sound quality has a higher weight. If we use these category features to generate product representations, we cannot weigh these categories based on the characteristics of the phone itself, because the weight distribution of different phones is very different. And a large number of category features will cause data sparseness. The short description is what the merchant claims to be selling, but the users may not necessarily agree, and the user's concerns are not completely included in this description. Thus, it is a good choice to use encoded-reviews as product representations. For a smartphone, from its historical reviews, we can find out which attribute characteristics of it will affect users' purchases of it, such as "The aspect ratio of the mobile phone is strange, making the entire screen look a little ugly," It can be known that the ratio of screen length to width is an important factor affecting customers' purchase of the phone. Extracting these important attribute features and user preferences from multiple historical reviews of this product can more accurately characterize the product features.

As we have mentioned before, E-Commerce platforms divide their products into several domains to satisfy the need of users. Users could buy products from different domains and leave review texts and overall ratings. These reviews and ratings provide the possibility to solve User Cold Start Problem from the perspective of Cross-Domain Recommendation [7]. Considering cold-start users in the target domain, although the interactive information of them and the products in the target domain is missing, they may leave some overall rating and review text information in some other domain behind. There is some user preference information that the product recommender system could learn to transform into the target domain, which may benefit the recommendation list of the target domain's products.

Some of the historical reviews are important, and some are harmful. Before extracting preference features for products and users, we first select the reviews. That is, pick the most critical reviews for each user and product pair.

In this paper, a review-aware cross-domain recommendation algorithm, called RACRec, is proposed. Firstly, a CNN [8] is employed to encode each original review into a vector, which incorporates the sentiment information of the review text. Secondly, valuable reviews are selected for each user and product pair. Thirdly, domain-specific preference vectors in the source domain and target domain and domain-shared preference vectors in the source domain and target domain of the cold start user are extracted by a migration model. On the other hand, the product feature vector in the target domain is generated from review texts by encoder and decoder. The product feature vector is combined with preference vectors of the cold-start user to make the rating prediction. The contributions of this paper are summarized as follow:

1) A cross-domain recommendation algorithm RACRec is proposed to generate the prediction for fully-cold-start users in product recommendation.

2) A cross-domain migration model of reviews is used to generate preference features for fully-cold-start users.

3) A strategy of review selection is implemented to select useful historical cross-domain reviews dynamically for each user and product pair.

4) Experiments have been designed to validate and analyze the performance of RACRec both on cold and non-cold data.

The rest of this paper is structured as follows. Section II will introduce some related work. The detailed structure of RACRec will be explained in Section III, followed by experimental results on the Amazon Dataset in Section IV. The conclusion of this paper is presented in Section V.

II. RELATED WORK

A. DEEP LEARNING RECOMMENDER SYSTEM

Deep learning due to its power to form the high quality of the user/item representation has been applied in the research of the recommender system widely. CNN has been widely used for feature representation and has achieved good results [18]–[20]. DeepCoNN [18] constructs the user and item representation by two parallel CNN structure, which consumes the review text. The representations of user and item are put into Factorization Machine to predict the overall rating. However, it's a single domain recommendation algorithm, which is not suitable for the cold-start user. SentiRec [19] incorporates the sentiment information of review text when modeling the user preference and product feature, which is more scalable. MV-DNN [21] model is proposed to map users and items to a latent space where the similarity between users and their preferred items is maximized.

B. REVIEW-AWARE RECOMMENDER SYSTEM

The review text written by a user for a product contains rich information such as the description of the preference of the user, the feature of the product, and how much the user likes it. By integrating such information, a recommender system could enhance the effectiveness of recommendation. Many works have mining such information to enhance the precision of recommendation. Text mining is an ascent technology in NLP. It is popular to leverage the topic model technology to extract the topic information from review text, which is combined with the latent factor model to form a more personalized recommendation, just like [9], [10]. Another branch is to mining the sentiment information from review text to make the user profile more accurately and an interpretable recommendation, just like [11]–[13]. The probability generation model is also integrated with review text to improve the precision of recommendation as well as to integrate the interpretable of recommendation, such [14], [15]. Some research focuses on extracting what aspect a review text is talking about. Such aspect information is combined with the latent factor model [16] or generative model [17] to make the rating prediction.

When recommending product i to user u, historical reviews are not equally important, some are useful, and some are useless, so we choose the most relevant reviews for this pair (u, i). The correlation between two vectors can be learned through Co-attentions [31], [32], and the corresponding Co-attentions weights can be obtained. Affinity Matrix is one of the effective ways to characterize Co-attentions [33].

C. CROSS DOMAIN RECOMMENDER SYSTEM

The cross-domain recommendation is considered as a practical approach to address the cold-start problem for its ability to introduce extract information about users or items. Early approaches in the cross-domain recommendation are to extend the traditional recommendation algorithm such as Matrix Factorization, Collaborative filtering by simply merging additional datasets, which didn't consider the heterogeneous between different datasets [22]. Later, the approaches to learning the mapping relationship between user and item are popular by which the user preference information or item feature information could be transformed from the source domain to the target domain, such as [23].

To solve the cold start problem, some models use only rating information for cross-domain migration [26]–[28]. LSCD [26] generates shared user features and domain-specific user features for rating matrices in multiple domains. CoNet [27] transfers user characteristics to address cold start problems by using neural networks as the base model. EMCDR [28] uses the scoring matrix to extract features of users and items and then employs a deep neural network to migrate features between domains. They only use the rating information and do not consider the more informative review information so that the effect will be worse in the case of a fully-cold-start. Besides, they poorly perform when recommending products with a little history to users, because there is too little rating information available.

There are also some cross-domain recommendation algorithms that use reviews to connect different domains, that is, reviews of users in different domains are used to extract shared features between domains. For example, to solve the "out-of-vocabulary" problem of cross-domain transfer, Cen *et al.* [29] extract a new CNN based model that leverages both word-level and character-based representations. RC-DFM [30] uses SDAE (stacked denoising autoencoder) to combine reviews, content information, and ratings to make cross-domain recommendations, and uses source domain reviews and product content information to alleviate cold-start issues. But they have a common problem that all the review information is migrated from the source domain without filtering.

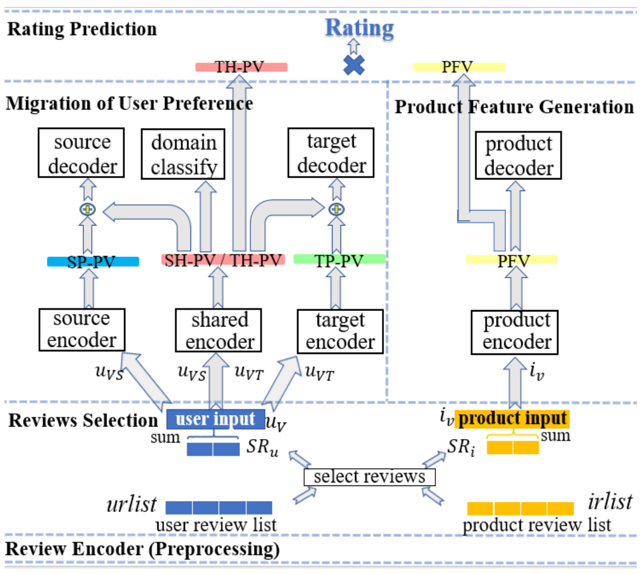


FIGURE 1. The overall architecture of RACRec.

This method will introduce noise because of the gap between the domains.

The latest cross-domain recommendation algorithm I-DSN [24] uses DSN (Domain Separation Network) [25] to extract User Vector u, and SDAE is used to extract item hidden representation h. The final classification result is corrected by minimizing the difference between v ($\mathrm{v} = [\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_L]$) and h, where v is the softmax weights of u. Assuming that there are L items in the target domain, then the discrete content features of these L items are input to SDAE to obtain h, and the discrete content features x of the items purchased by the user in the source and target domains are used as the input of DSN. Finally, get the prediction result y ($\mathrm{y} = [\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_L]$), where y_k represents the probability of the user purchasing item k. The calculation process of y is as (1).

$$\mathrm{y}_k = \mathrm{p}\left(\mathrm{y} = \mathrm{k} \mid \mathrm{x}^T\right) = \frac{\exp(\mathrm{v}_k^T \mathrm{u})}{\sum_{k' \in [1,2,\ldots,L]} \exp(\mathrm{v}_{k'}^T{}_{\mathrm{u}})} \quad (1)$$

In the I-DSN, the feature representation h of the items is only used to modify the softmax weight of the user feature and is not directly used to calculate the prediction result, which causes the model to focus on the user feature. But the item feature learning is not sufficient. However, the feature generation part of the user and product in our model is parallel, product features and user features are equally important for the prediction.

III. METHODOLOGY

The overall architecture of RACRec is shown in Fig.1. It composes of 'Review Encoder' for encoding the original review into a vector with sentiment information, 'Reviews Selection' for selecting significant reviews for a pair of user and product, 'Migration of User Preference' for transforming the user preference information from the source domain to target domain, 'Product Feature Generation' for compressing

TABLE 1. Symbols in the paper and their meaning.

Symbol	Descriptions
d	the length of each word vector in an original review
l	the number of units of the fully connected layer in the reviews encoder
u	the user
i	the product
k	the number of convolution kernels in the reviews encoder
Z	the number of historical reviews for users and products
K	the number of reviews selected from Z reviews
ε_{pred}	the coefficient of L_{pred}
E_{SP}	source encoder of the user
E_{SH}	shared encoder of the user
E_{TP}	target encoder of the user
E_{I}	encoder of the product in the target domain
D_{SP}	source decoder of the user
D_{TP}	target decoder of the user
D_{I}	decoder of the product in the target domain
SH-PV	user's domain-shared preference vectors in the source domain
SP-PV	user's domain-specific preference vectors in the source domain
TH-PV	user's domain-shared preference vectors in the target domain
TP-PV	user's domain-specific preference vectors in the target domain
PFV	product's feature vectors in the target domain

the vector representation of product and 'Rating Prediction' for taking the preference information of cold-start user and feature information of product to generate the rating. The main symbols used in this paper and their descriptions are given in Table 1.

A. REVIEW ENCODER

It is a pre-processing process that reduces time and space complexity. And users usually use sentiment words to express whether they love a product or not. By mining such information, a recommender system could build the user profile more accurately, which means a more personalized recommendation list. We also argue that emotional representations are suitable for product representations. Because in the reviews, users describe the preferences for specific attributes of the products in a fine-grained way. Therefore, the features extracted by using the sub-network include not only the attribute characteristics of the product but also the emotional information of historical users on these attributes. When recommending the product to other users, such product characteristics can more effectively represent the product. While, in the modern E-Commerce platform, review-text feedback always accompanies with overall rating feedback. Usually, emotion user expresses in the review text, and the overall rating is consistent. A natural inference is that the overall rating information could be thought of as the emotion label of the review text. Here, we define the emotion label of review

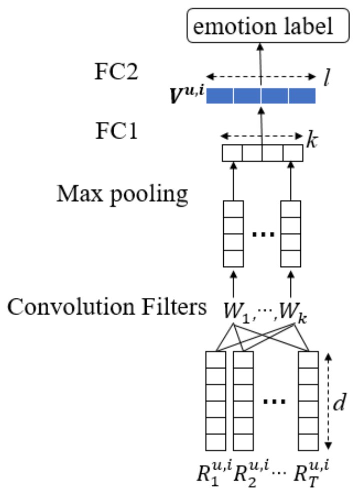


FIGURE 2. The process of reviews encoding.

text as (2) shows.

$$emotion = \begin{cases} 1, & if\ overall\ rating \geq 3 \\ 0, & if\ overall\ rating < 3 \end{cases} \tag{2}$$

The emotion of review text with a rating greater or equal to 3 is regarded as positive. Inspired by SentiRec [19], we set up a CNN to incorporate such sentiment information when encoding each review text into vector form. The architecture is shown in Fig.2.

Each original review text $R^{u,i}$ related to user u and product i is consumed by several convolution filters W ($W = [W_1, \ldots, W_k]$), and the window size of *n-th* convolution filter $W_n \in R^{d\times s}$ is s. Then, Max Pooling is arranged after the Convolution Layer to capture the most import feature. To get a vector $V^{u,i}$ for the review, we next use a fully connected layer FC1 with l units. In order to incorporate the sentiment information into such a vector, Logistic Regression is delegated to predict the emotion label of the review text by taking the vector $V^{u,i}$ as input. The parameter of this structure is updated by minimizing the cross-entropy loss of the emotion label prediction. Finally, the output of the fully connected layer is saved as the vector form of every review text.

B. REVIEWS SELECTION

For a pair of user u and product i, the most recent Z reviews are used as input. The user review list *urlist* is spliced by u's review vectors, i.e., $urlist = [V^{u,1}, V^{u,2}, \ldots, V^{u,Z}]$, and the product review list *irlist* is stitched together by i's review vectors, i.e., $irlist = [V^{1,i}, V^{2,i}, \ldots, V^{Z,i}]$. Our goal is to select the most important K reviews for user and item, respectively.

1) CALCULATION OF AFFINITY MATRIX

The Affinity Matrix A is calculated as (3).

$$A_{p,q} = F\left(urlist_p\right) \cdot M \cdot F\left(irlist_q\right), \tag{3}$$

where $M \in R^{l\times l}$ and $A \in R^{Z\times Z}$. $F(.)$ is a feed-forward neural network function with f layers and l units.

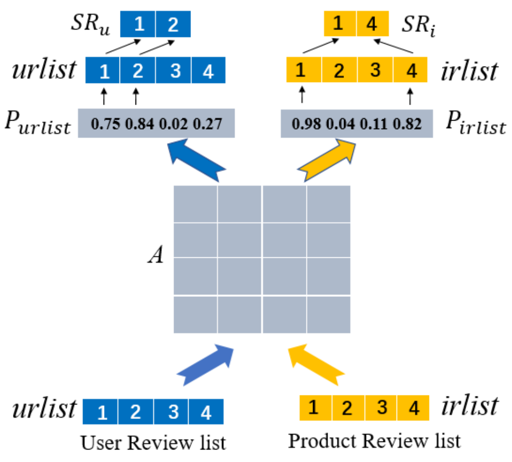


FIGURE 3. **The process of the reviews selection.**

2) REVIEW POINTERS

By taking the maximum of rows and columns of the matrix A, co-attentions weights for *urlist* and *irlist* is calculated as (4).

$$W_{urlist} = max_{col}(A) \quad \text{and } W_{irlist} = max_{row}(A), \tag{4}$$

where W_{urlist} is Co-attentions weight of *urlist*, and W_{irlist} is Co-attentions weight of *irlist*.

The index of the largest K weight values in W_{urlist} is taken as a pointer P_{urlist} to *urlist*. The index of the largest K weight values in W_{irlist} is taken as a pointer P_{irlist} to *irlist*, which is shown as (5).

$$\begin{aligned} P_{urlist} &= W_{urlist}.index\,(max\,(W_{urlist}, K)) \\ P_{irlist} &= W_{irlist}.index\,(max\,(W_{irlist}, K)) \end{aligned} \tag{5}$$

Then the corresponding review vectors SR_u is selected from *urlist* according to the pointer P_{urlist}, and SR_i is selected from *irlist* according to the pointer P_{irlist}. The vector we get above is just a review-wise representation. To form the user or product representation (u_v or i_v), we need to average all the review vectors one has, just as (6). The process is shown in Fig.3.

$$\begin{aligned} u_v &= \frac{\sum_{k=1}^{K} SR_u^k}{T} \\ i_v &= \frac{\sum_{k=1}^{K} SR_i^k}{T} \end{aligned} \tag{6}$$

C. MIGRATION OF USER PREFERENCE MODEL

This paper is based on the assumption that user preference information in one domain is consists of domain-shared part and domain-specific part. Therefore, the user preference vector in the source domain is divided into User Domain-Specific Preference Vector (SP-PV) and Domain-Shared Preference Vector (SH-PV). Similarly, the user preference vector in the target domain is also divided into domain-specific one (TP-PV) and domain-shared one (TH-PV).

The approach to migrate the cold-start user's preference information from the source domain to the target domain we used is to take the cold-start user's SH-PV as his preference vector TH-PV in the target domain. Inspired by Domain Separation Network [25], this paper has designed an architecture to accomplish this task, which can be seen in the Migration of User Preference of Fig.1.

The whole architecture is an encoder-decoder structure. The encoder part consists of source encoder, target encoder, and shared encoder, whose parameters are independent. The user representation (u_v) from different domains will be feed into source encoder and target encoder separately to generate the representation of the user's specific preference information in the current domain. As for the shared encoder, it will consume all u_v from every domain to extract the user preference information, which could be shared among every domain.

As shown in Fig.1, u_{VS} is the user input from the source domain and u_{VT} is the user input from the target domain. Let $E(\cdot)$ as the encoder function, and $D(\cdot)$ as the decoder function. The shared encoder E_{SH} and source encoder E_{SP} generate Ss and Sp separately by consuming the same u_{VS}. Similarly, Ts and Tp is constructed by shared encoder E_{SH} and target encoder E_{TP} by taking u_{VT} as input, which is shown in (7).

$$\begin{aligned} S_s &= E_{SH}(u_{VS}) \quad \text{and } S_p = E_{SP}(u_{VS}) \\ T_s &= E_{SH}(u_{VT}) \quad \text{and } T_p = E_{TP}(u_{VT}) \end{aligned} \tag{7}$$

Domain-specific and domain-shared features should be easily distinguishable, so the difference between Ss and Sp should as large as possible, as are Ts and Tp. The loss of difference has been set up as (8).

$$L_{diff} = \left\| S_s^T \cdot S_p \right\|_F^2 + \left\| T_s^T \cdot T_p \right\|_F^2 \tag{8}$$

Besides, the target of shared encoder is to extract the user preference vector, which could be shared between domains, which means it can't identify the domain from which it is trained. To achieve such effect, Logistic Regression Domain Classification is employed to predict the domain label y of Ss (or Ts). For the output x of shared encoder ($x = E_{SH}(input)$), if *input* belongs to the source domain, the label y of x is 1. Otherwise, the label is 2. By maximizing the domain classification loss caused by Logistic Regression Domain Classification, the output of shared encoder from a different domain is encouraged to be more distinguishable, which means the output is similar enough so that the user domain-shared preference vector from source domain could be migrated to the target domain. The loss of Logistic is shown as (9).

$$L_{class} = -\left[I_{y=1} log\,(p\,(y = 1\,|\,x)) + I_{y=2} log\,(p\,(y = 2\,|\,x)) \right] \tag{9}$$

Here, $I_{y=1}$ is the indication function. It equals 1 when y equals 1. And $p(y = 1|x)$ is the probability that the user domain-share preference vector is x, and the predicted label is 1.

On the other hand, to ensure that the SH-PV (or TH-PV) and SP-PV (or TP-PV) are precise enough, a reconstruction loss is added by introducing two domain-specific decoders.

Recall that user preference information in each domain is consists of domain-shared part and domain-specific part. The domain-shared part and domain-specific part generated by domain-shared encoder and domain-specific encoder are added together. Such vectors are consumed by the domain-specific decoder to reconstruct the user input. The reconstructed user input could be represented as (10).

$$\begin{aligned} u'_{VS} &= D_{SP}\left(S_s + S_p\right) \\ u'_{VT} &= D_{ST}\left(T_s + T_p\right), \end{aligned} \quad (10)$$

where D_{SP} is the domain-specific decoder of the source domain, D_{ST} is the domain-specific decoder of the target domain.

So the reconstruction loss could be introduced as (11).

$$L_{rec} = \left\|u_{VS} - u'_{VS}\right\|^2 + \left\|u_{VT} - u'_{VT}\right\|^2 \quad (11)$$

D. PRODUCT FEATURE AND RATING PREDICTION

Here, we employ an encoder-decoder model to generate product features. Let $E_I(\cdot)$ be an encoder function, $D_I(\cdot)$ be a decoder function, and i_V is the input of E_I in the target domain.

I_p is the product feature calculated by (12), and L_{irec} is the reconstruction loss could be expressed as (12).

$$\begin{aligned} I_p &= E_I(i_V) \\ i'_V &= D_I(I_p) \\ L_{irec} &= \left\|i_V - i'_V\right\|^2 \end{aligned} \quad (12)$$

It is not enough to predict whether a user will buy a product. We also need to predict the user's rating on the product. Taking Amazon as an example, users will score the purchased products, and the score is in [1]–[5]. Score 1 indicates that the user does not like this product, and score 5 indicates that he likes it. For the user, he gave the product i a score of 1. The score indicates that the user will click or buy the product, but a score of 1 indicates that he does not like the product, so recommending i to the user is a failure. Therefore, if the purchase is only used as the evaluation indicator, user satisfaction will decrease. So we divide the recommendation task into 1) judging whether the user purchases the product and 2) predicting the user's rating on the product.

When predicting whether a user purchases a product, the entire algorithm is a classification task, and the prediction result P is calculated by (13). Negative log loss is used to measure the gap between the predicted result P_i and the true label Tr_i. The objective function of the prediction part is shown in (14).

$$P = softmax(Dense([T_s, I_p])), \quad (13)$$

where *Dense* represents the fully connected layer.

$$L_{pred} = \sum_{i=1}^{num} -P_i \cdot \log(Tr_i) \quad (14)$$

When predicting the rating, the algorithm is a regression task. The overall rating for u and i could be predicted as (15)

$$rating_{pred} = T_s \cdot I_p \quad (15)$$

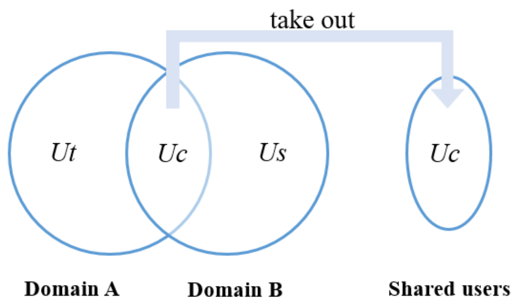


FIGURE 4. The relationship of users in domain A and domain B. We keep U_c only.

The difference between the prediction rating and true rating is introduced as the loss of prediction as (16).

$$L_{pred} = \sqrt{\frac{\sum_{i=1}^{num}\left(rating_{pred_i} - rating_{true_i}\right)^2}{num}} \quad (16)$$

E. OBJECTIVE FUNCTION

As we have introduced above, RACRec has several sub-loss functions, including L_{diff}, L_{class}, L_{rec}, L_{irec}, and L_{pred}. When training RACRec, we set up the total objective function as (17).

$$L = L_{diff} + L_{class} + L_{rec} + L_{irec} + \varepsilon_{pred} \cdot L_{pred}, \quad (17)$$

ε_{pred} is the coefficient of L_{pred}.

IV. EXPERIMENTS

Experiments are designed to explore the performance of RACRec. As for RACRec addresses Fully User Cold Start Problem in a cross-domain way, the experiments in this section are on a pair of domains in the Amazon Dataset. One for source domain, while the other as the target domain. The relationship of users in the two domain is shown as Fig.4. U_t is the user who only have review texts and overall rating information in Domain A, while history information of U_s could only be found in Domain B. And the review text and overall rating information of U_c could be found in both domains. This paper uses only U_c to construct the dataset.

The Amazon Dataset is collected by Julian McAuley [25], [19], which contains 142.8 million reviews and overall rating spanning 1996.05∼2014.07. Each piece of data contains *uid*, *iid*, *rating*, and *review*, where *uid* is the user ID, *iid* is the product ID, *rating* is the overall rating, and *review* is the text that the user wrote for the product.

A. EXPERIMENT SETUP

This paper selects three pairs of source and target domains, which are: Electronics (EI) & Cell_Phones_and_Accessories (cell), EI & Sports_and_Outdoors (sports), EI & Video_Games (video). The reason for choosing EI as the source domain is that EI is a diversity domain which is large enough, and there are many overlapping users in EI and other domains. Three domains (cell, sports, video) which share the most common users with EI, are selected as target domains.

TABLE 2. Numerical statistics of U25_I25_Cold and U25_I25_NonCold.

		U25_I25_Cold		U25_I25_NonCold	
		train	test	train	test
sport	user	155	38	193	193
	item	4149	1282	4134	1252
	reviews	6524	1454	6310	1668
video	user	132	33	165	165
	item	4240	1130	4061	1256
	reviews	7897	1419	7386	1930
cell	user	119	29	148	148
	item	1973	853	2086	683
	reviews	5697	1213	5473	1437

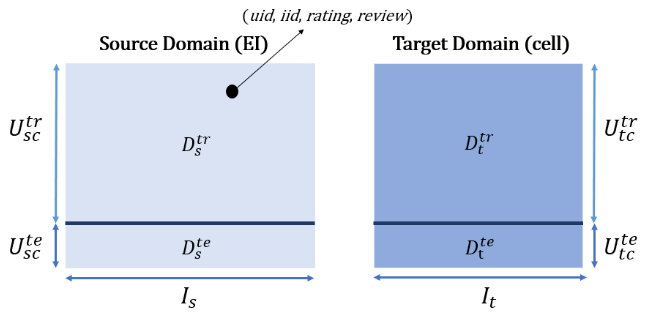


FIGURE 5. The process of forming U25_I25_Cold. $U_{sc}^{tr} = U_{tc}^{tr}$, $U_{sc}^{te} = U_{tc}^{te}$, $D_s = D_s^{tr} + D_s^{te}$, $D_t = D_t^{tr} + D_t^{te}$, $U_c = U_{tc}^{tr} + U_{tc}^{te} = U_{sc}^{tr} + U_{sc}^{te}$.

To simulate a fully-cold-start situation, we generate a fully-cold-start dataset U25_I25_Cold and a non-cold-start dataset U25_I25_NonCold. The statistics of U25_I25_Cold and U25_I25_NonCold are shown in Table 2. 'U25_I25' means that we select the data with more than 25 records of *uid* in EI and cell, and more than 25 reviews of *iid* in EI (or cell). Data set D_s of EI and data set D_t of cell are then generated.

U25_I25_Cold: As in Fig.5, EI and cell have the same users U_c. U_c is divided into 4: 1 to get U_{sc}^{tr} (U_{tc}^{tr}) and U_{sc}^{te}(U_{tc}^{te}). I_s and I_t are the product sets of EI and cell, respectively. Then we get the training set D_s^{tr} and the testing set D_s^{te} in EI, the training set D_t^{tr} and the testing set D_t^{te} in cell. The reviews of U_{tc}^{tr} in D_t^{tr} are removed, we only use their review information in D_s^{tr} to train our model, and we can only use historical reviews of U_{tc}^{te} in D_s^{te}.

U25_I25_NonCold: We take the first 80% records of each user in D_s as the training set and the last 20% as the testing set, and do the same for the cell. Users can use two kinds of information, which come from EI and cell.

We set $l = 50$, $d = 100$, $k = 26$, $s = 4$, $Z = 100$, $f = 1$, and $\varepsilon_{pred} = 1000$. All encoder ($E_{SP}$, E_{SH}, E_{ST}, E_I) are composed of two fully connected layers, and the number of units of which is [25], [12]. All decoders (D_{SP}, D_{ST}, D_I) are composed of two fully connected layers, and the number of units of which is [25], [50]. We use the grid search to determine the optimal parameters. Each parameter has a series of values, such as K is searched in the interval [0, 10, 50, 80], ε_{pred} is searched in the interval [0.1, 1, 10, 100, 1000, 10000, 100000] and so on. Finally, the optimal parameter combination on each data set is determined based on the results of the grid search.

B. BASELINE ALGORITHM

Some baseline algorithms we compare to are described as follows.

1) CoNet [27]

A cross-domain recommendation algorithm based on neural network to implement feature migration. We actualize CoNet with python code.

2) LSCD [26]

A Cross-Domain Collaborative Filtering algorithm. LSCD also extracts domain-specific and domain-shared features for users. The difference is that these features are extracted through rating matrixes. We implement it using the matlab code provided by the author.

3) BiasedMF

A traditional recommendation algorithm, which predicts the overall rating as the product of the latent user vector and latent product vector.

4) SVD++

The variant of BiasedMF, which considers the appearance of user-product interaction as an indication of user preference to generate more precision recommendation.

5) I-DSN [24]

I-DSN is a cross-domain recommendation algorithm, which uses DSN to migrate user preference across domains.

C. EXPERIMENT RESULT

We consider two evaluation indicators, AUC and RMSE. The smaller the RMSE, the better the effect. The bigger the AUC, the better the effect. When the evaluation indicator is AUC, we sample 99 negative examples for each user for training and prediction. AUC represents the probability that the predicted positive example is ranked before the negative example. For the task of determining whether a user purchases a product, AUC is used to evaluate the recommendation effect.

The rating prediction problem considers how to predict the rating accurately, so we use RMSE to evaluate the prediction effect, as shown in (18).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{len(test)} (rating_{pred_i} - rating_{true_i})^2}{len(test)}}, \quad (18)$$

where *test* is the testing set.

The best performance of these algorithms is shown in boldface. The experiment results of U25_I25_Cold is presented in Table 3. We can find that RACRec achieves the smallest RMSE in all domains. The improvement of RMSE

TABLE 3. RMSE and AUC results of U25_I25_Cold.

		RMSE			AUC		
		sport	video	cell	sport	video	cell
single domain	MF	0.8276	0.9893	0.8834	0.5054	0.5444	0.5274
	SVD++	0.8296	1.0003	0.8910	-	-	-
cross domain	I-DSN	0.8311	1.0077	0.8986	0.5298	0.5382	0.5608
	CoNet	0.8297	0.9862	0.8814	0.5721	0.6155	0.6688
	LSCD	0.8305	1.0061	0.8972	-	-	-
	RACRec	**0.8228**	**0.9806**	**0.8784**	**0.6246**	**0.7134**	**0.6751**
	Imp	**0.58%**	**0.57%**	**0.34%**	**9.18%**	**15.91%**	**0.94%**

TABLE 4. RMSE and AUC results of select reviews of U25_I25_Cold.

	RMSE			AUC		
Target	sports	video	cell	sports	video	cell
CoNet	0.8297	0.9862	0.8814	0.5721	0.6155	0.6688
RACRec_sele50	0.8245	0.9840	0.8807	0.6239	0.7122	0.6715
RACRec_sele10	**0.8228**	**0.9806**	**0.8784**	**0.6246**	**0.7134**	**0.6751**
RACRec_sele0	0.8298	1.0013	0.8956	0.5618	0.7119	0.6688

TABLE 5. The impact of each loss on the recommendation effect.

L_{rec}	L_{diff}	L_{class}	L_{irec}	L_{pred}	RMSE
√	√	√	√	√	**0.9792**
×	√	√	√	√	0.9812
√	×	√	√	√	0.9863
√	√	×	√	√	0.9806
√	√	√	×	√	0.9821
√	√	√	√	×	1.8792

√ represents the loss is not 0, × represents the loss is 0

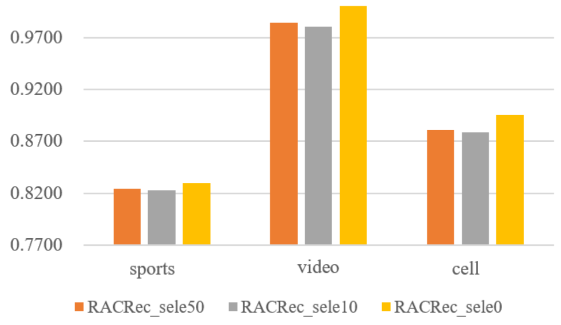


FIGURE 6. Effect of review selection on RMSE.

of RACRec on sports, video, and cell is 0.58%, 0.57%, and 0.34%. In the case of fully-cold-start users, the best performing baseline model is CoNet. However, LSCD performs poorly, even worse than the simplest matrix factorization because it only extracts features based on matrix factorization. I-DSN performs poorly because its product characteristics are not directly involved in the prediction of results. RACRec achieves the largest AUC in all domains. The improvement of AUC of RACRec on sports, video, and cell is 9.18%, 15.91%, and 0.94% in the case of fully-cold-start users. Our RACRec learns the characteristics of users and products from reviews. Compared with the rating, the advantage of the review becomes apparent, which can more accurately and meticulously capture the interest characteristics of users.

1) THE ROLE OF SELECTED REVIEWS

Table 4 shows the comparison of the results of CoNet and the three forms of and RACRec, which including RACRec_sele50, RACRec_sele10, and RACRec_sele0. RACRec_sele50 means that we select the 50 most important reviews from user's and product's historical reviews, respectively. RACRec_sele0 means that we do not select reviews, and all reviews are retained. It can be seen from Fig.6 and Fig.7 that choosing a different number of reviews will have a great impact on the recommendation result, and the ranking is RACRec_sele10 > RACRec_sele50 > RACRec_sele0. On sports, cell, and video, selecting the most relevant 10 ones is more appropriate. The effect is very bad if no selection is made at all because there are too much noise data across the domain. Too many reviews are selected will also reduce the effect, and choosing the proper number of reviews is the best.

2) EFFECT OF EACH LOSS

In the training process, the five parts of the loss function will affect the parameter update, so the model will not focus only on the loss of the RMSE part. To verify the above statement, we set the five parts of the loss to 0 one by one to illustrate the effect of each part on the recommendation result, and the results are in Table 5. The loss consists of

TABLE 6. RMSE and AUC results of U25_I25_NonCold.

	RMSE			AUC		
target	sports	video	cell	sports	video	cell
CoNet	**0.8481**	**0.9788**	**0.7748**	**0.7112**	**0.7786**	**0.7610**
RACRec_sele50	0.8889	1.0580	0.8267	0.6118	0.6291	0.5860
RACRec_sele10	0.8901	1.0475	0.8337	0.6498	0.6477	0.5995
RACRec_sele0	0.8902	1.0627	0.8206	0.3703	0.2966	0.2773

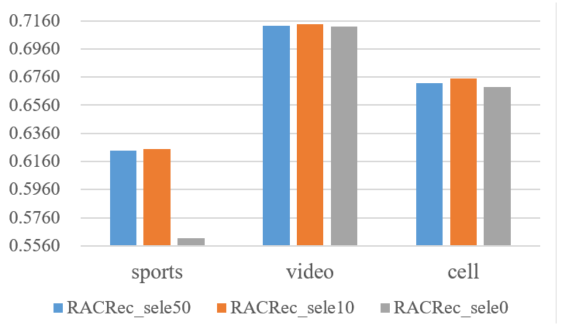


FIGURE 7. Effect of review selection on AUC.

L_{diff}, L_{class}, L_{rec}, L_{irec}, and L_{pred}, where L_{pred} is the RMSE loss. It can be seen from Table 5 that any loss set to 0 will reduce the recommended effect, and when L_{pred} is set to 0, the effect reduces the most. The above result is in line with our expectations. Our main task is to predict the rating. The other part of the loss is to serve the feature construction in the prediction process.

3) EFFECT ON NON-COLD START DATA

We also performed experiments on the non-cold start dataset U25_I25_NonCold, as shown in Table 6. When users and products have richer information in both domains, the effect of CoNet becomes better. At this time, the review selection also loses its utility. The possible reason is that the historical data in the current domain is useful and low noise. So RACRec_sele0 without feature selection will be more suitable than RACRec_sele50 and RACRec_sele10 in cell.

V. CONCLUSION

In this paper, we propose a cross-domain recommendation algorithm RACRec for Fully User Cold Start Problems and make a valid selection for cross-domain reviews. Experiments have proven that RACRec works well, and the review selection is indeed indispensable. However, RACRec does not solve the non-cold start problem very well. In the future, we will consider introducing CNN into the process of user feature migration and migrate rating and review information at the same time.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant 61976239, in part by the Guangzhou Science and Technology Program under Grant 201903010046, and in part by the Zhongshan Innovation Foundation of High-End Scientific Research Institutions under Grant 2019AG031.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[2] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[3] W. Luan, G. Liu, C. Jiang, and L. Qi, "Partition-based collaborative tensor factorization for POI recommendation," *IEEE/CAA J. Automatica Sinica*, vol. 4, no. 3, pp. 437–446, Jul. 2017.

[4] W. Luan, G. Liu, and C. Jiang, "Collaborative tensor factorization and its application in POI recommendation," in *Proc. IEEE 13th Int. Conf. Netw., Sens., Control (ICNSC)*, Apr. 2016, pp. 1–6.

[5] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 2009, Aug. 2009, Art. no. 421425.

[6] L. H. Son, "Dealing with the new user cold-start problem in recommender systems: A comparative review," *Inf. Syst.*, vol. 58, pp. 87–104, Jun. 2016, doi: 10.1016/j.is.2014.10.001.

[7] C. Iván, F. Ignacio, B. Shlomo, and C. Paolo, *Cross-Domain Recommender Systems. Recommender Systems Handbook*, 2nd ed. Springer, 2015, pp. 919–959, doi: 10.1007/978-1-4899-7637-6_27.

[8] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*. [Online]. Available: http://arxiv.org/abs/1408.5882

[9] W. Zhang and J. Wang, "Integrating topic and latent factors for scalable personalized review-based rating prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 11, pp. 3013–3027, Nov. 2016, doi: 10.1109/TKDE.2016.2598740.

[10] H. Wang, Y. Lu, and C. Zhai, "Latent aspect rating analysis without aspect keyword supervision," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2011, pp. 618–626, doi: 10.1145/2020408.2020505.

[11] L. Chen and F. Wang, "Explaining recommendations based on feature sentiments in product reviews," in *Proc. 22nd Int. Conf. Intell. User Interface (IUI)*, 2017, pp. 17–28, doi: 10.1145/3025171.3025173.

[12] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, "Explicit factor models for explainable recommendation based on phrase-level sentiment analysis," in *Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2014, pp. 83–92, doi: 10.1145/2600428.2609579.

[13] Q. Diao, M. Qiu, C.-Y. Wu, A. J. Smola, J. Jiang, and C. Wang, "Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS)," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2014, pp. 193–202, doi: 10.1145/2623330.2623758.

[14] J. McAuley and J. Leskovec, "Hidden factors and hidden topics: Understanding rating dimensions with review text," in *Proc. 7th ACM Conf. Recommender Syst. (RecSys)*, 2013, pp. 165–172, doi: 10.1145/2507157.2507163.

[15] L. Guang, R. L. Michael, and K. Irwin, "Ratings meet reviews, a combined approach to recommend," *Proc. 8th ACM Conf. Recommender Syst. (RecSys)*, 2014, pp. 105–112, doi: 10.1145/2645710.2645728.

[16] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli, "Aspect-aware latent factor model: Rating prediction with ratings and reviews," in *Proc. World Wide Web (WWW)*, 2018, pp. 639–648, doi: 10.1145/3178876.3186145.

[17] H. Li, R. Lin, R. Hong, and Y. Ge, "Generative models for mining latent aspects and their ratings from short reviews," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2015, pp. 241–250, doi: 10.1109/ICDM.2015.28.

[18] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, 2017, pp. 425–433, doi: 10.1145/3018661.3018665.

[19] H. Dongmin, P. Chanyoung, Y. Min-Chul, S. Ilhyeon, L. Jung-Tae, and Y. Hwanjo, "Review sentiment-guided scalable deep recommender system," in *Proc. SIGIR*, vol. 18, Jun. 2018, pp. 965–968, doi: 10.1145/3209978.3210111.

[20] L. Zhenchuan, L. Guanjun, and J. Changjun, "Deep representation learning with full center loss for credit card fraud detection," *IEEE Trans. Computat. Social Syst.*, early access, Feb. 17, 2020, doi: 10.1109/TCSS.2020.2970805.

[21] A. M. Elkahky, Y. Song, and X. He, "A multi-view deep learning approach for cross domain user modeling in recommendation systems," in *Proc. 24th Int. Conf. World Wide Web (WWW)*, 2015, pp. 278–288, doi: 10.1145/2736277.2741667.

[22] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 2008, pp. 650–658, doi: 10.1145/1401890.1401969.

[23] R. He, C. Packer, and J. McAuley, "Learning compatibility across categories for heterogeneous item recommendation," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 937–942, doi: 10.1109/ICDM.2016.0116.

[24] K. Heishiro, K. Hayato, S. Nobuyuki, T. Yukihiro, and S. Taiji, "Cross-domain recommendation via deep domain adaptation," in *Proc. Eur. Conf. Inf. Retr.* Cham, Switzerland: Springer, 2019.

[25] B. Konstantinos, T. George, S. Nathan, K. Dilip, and E. Dumitru, "Domain separation networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 343–351.

[26] Z. Zhi-Lin, H. Ling, W. Chang-Dong, and H. Dong, "Low-rank and sparse cross-domain recommendation algorithm," in *Proc. Int. Conf. Database Syst. Adv. Appl.* Cham, Switzerland: Springer, 2018.

[27] G. Hu, Y. Zhang, and Q. Yang, "CoNet: Collaborative cross networks for cross-domain recommendation," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2018, pp. 667–676.

[28] T. Man, H. Shen, X. Jin, and X. Cheng, "Cross-domain recommendation: An embedding and mapping approach," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 2464–2470.

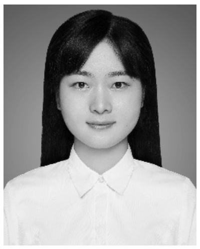
[29] C. Cen, Y. Yinfei, Z. Jun, L. Xiaolong, and B. Forrest Sheng, "Cross-domain review helpfulness prediction based on convolutional neural networks with auxiliary domain discriminators," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 2, 2018, pp. 602–607.

[30] W. Fu, Z. Peng, S. Wang, Y. Xu, and J. Li, "Deeply fusing reviews and contents for cold start users in cross-domain recommendation systems," *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 94–101.

[31] M. I. Hasan Chowdhury, K. Nguyen, S. Sridharan, and C. Fookes, "Hierarchical relational attention for video question answering," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 289–297.

[32] C. Xiong, V. Zhong, and R. Socher, "Dynamic coattention networks for question answering," 2016, *arXiv:1611.01604*. [Online]. Available: http://arxiv.org/abs/1611.01604

[33] Y. Tay, A. T. Luu, and S. C. Hui, "Multi-pointer co-attention networks for recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2309–2318.



YARU JIN was born in Yuncheng, China, in 1996. She received the B.S. degree in mathematics from the South China University of Technology, Guangzhou, China, in 2014, where she is currently pursuing the M.S. degree in computer science and engineering.

Her research interests include recommendation systems, transfer learning, and natural language processing.

SHOUBIN DONG was born in 1967. She received the B.S., M.S., and Ph.D. degrees in electronic engineering from the University of Science and Technology of China (USTC), Hefei, China, in 1994.

She was a Visiting Scholar with the School of Computer Science, Language Technology Institute, Carnegie Mellon University (CMU), Pittsburgh, USA, from 2001 to 2002. She is currently a Professor with the School of Computer Science and Engineering, South China University of Technology (SCUT), Guangzhou, China. Her research interests include information retrieval, natural language processing, and high-performance computing.

YONG CAI was born in Guangdong, China, in 1993. He received the B.S. and M.S. degrees from the South China University of Technology, Guangzhou, China, in 2019.

His research interests include recommendation systems, transfer learning, and natural language processing.

JINLONG HU received the B.S., M.S., and Ph.D. degrees from the South China University of Technology (SCUT), Guangzhou, China, in 2012.

He was a Visiting Scholar with Pennsylvania State University, USA, from 2016 to 2017. He is currently an Associate Professor with the School of Computer Science and Engineering, SCUT. His research interests include big data analysis and processing, machine learning, fraud detection, brain informatics, computational advertising and marketing, and next-generation network technology and security.