π-Net: A Parallel Information-sharing Network for Shared-account Cross-domain Sequential Recommendations

Muyang Ma[*]
Shandong University
Jinan, China
muyang0331@gmail.com

Pengjie Ren[*]
University of Amsterdam
Amsterdam, The Netherlands
p.ren@uva.nl

Yujie Lin[*]
Shandong University
Jinan, China
yu.jie.lin@outlook.com

Zhumin Chen
Shandong University
Jinan, China
chenzhumin@sdu.edu.cn

Jun Ma
Shandong University
Jinan, China
majun@sdu.edu.cn

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

Sequential Recommendation (SR) is the task of recommending the next item based on a sequence of recorded user behaviors. We study SR in a particularly challenging context, in which multiple individual users share a single account (shared-account) and in which user behaviors are available in multiple domains (cross-domain). These characteristics bring new challenges on top of those of the traditional SR task. On the one hand, we need to identify different user behaviors under the same account in order to recommend the right item to the right user at the right time. On the other hand, we need to discriminate the behaviors from one domain that might be helpful to improve recommendations in the other domains.

We formulate the *Shared-account Cross-domain Sequential Recommendation* (SCSR) task as a parallel sequential recommendation problem. We propose a **P**arallel **I**nformation-sharing **Net**work (π-Net) to simultaneously generate recommendations for two domains where user behaviors on two domains are synchronously shared at each timestamp. π-Net contains two core units: a *shared account filter unit* (SFU) and a *cross-domain transfer unit* (CTU). The SFU is used to address the challenge raised by shared accounts; it learns user-specific representations, and uses a gating mechanism to filter out information of some users that might be useful for another domain. The CTU is used to address the challenge raised by the cross-domain setting; it adaptively combines the information from the SFU at each timestamp and then transfers it to another domain. After that, π-Net estimates recommendation scores for each item in two domains by integrating information from both domains.

To assess the effectiveness of π-Net, we construct a new dataset HVIDEO from real-world smart TV watching logs. To the best of our knowledge, this is the first dataset with both shared-account and cross-domain characteristics. We release HVIDEO to facilitate future research. Our experimental results demonstrate that π-Net outperforms state-of-the-art baselines in terms of MRR and Recall.

[*]Joint first author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGIR '19, July 21–25, 2019, Paris, France
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331200

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Shared-account recommendation, Cross-domain recommendation, Sequential recommendation

ACM Reference Format:
Muyang Ma, Pengjie Ren, Yujie Lin, Zhumin Chen, Jun Ma, and Maarten de Rijke. 2019. π-Net: A Parallel Information-sharing Network for Shared-account Cross-domain Sequential Recommendations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, Article 4, 10 pages. https://doi.org/10.1145/3331184.3331200

1 INTRODUCTION

In *Sequential Recommendation* (SR) a recommender system has to promote recommendations based on a sequence of logged user behaviors, where interactions are usually grouped by virtue of taking place within the same time frame [11, 27, 29, 36, 38]. Users usually have specific goals during the process, such as finding a good restaurant in a city, or listening to music of a certain style or mood [35]. SRs have a wide range of applications in many domains such as e-commerce, job websites, music and video recommendations [39].

In this paper, we study SR in a particularly challenging context, *Shared-account Cross-domain Sequential Recommendation* (SCSR), in which multiple individual users share a single account (shared account) and in which user behavior is recorded in multiple domains (cross-domain). The shared account characteristic is considered because in some applications, people tend to share a single account. For example, in the smart TV recommendation scenario depicted in Figure 1, members of a family share a single account to watch videos. The existence of shared accounts makes it harder to generate relevant recommendations, because multiple user behaviors are mixed together.

We consider the cross-domain task because it is a common phenomenon in practice. Users use different platforms to access domain-specific services in daily life. We can get user behavior data from different domains during the same time period. User behaviors in

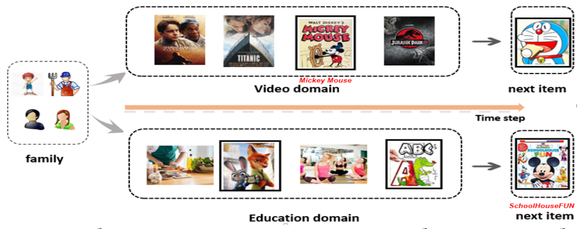

Figure 1: The smart TV scenario provides a natural example of *Shared-account Cross-domain Sequential Recommendation* (SCSR). Here, the *video* domain contains various movies, TV series, cartoons, etc. The *education* domain contains educational programs and technical tutorials, etc. $\boxed{\textit{Boxed}}$ items reflect similar user interests.

one domain might be helpful for improving recommendations in another domain [13, 21, 22, 40, 44], the idea being that user behavior in different domains might reflect similar user interests. For example, as shown in Figure 1, videos like "Mickey Mouse" in the video domain might help to predict the next item "Schoolhouse Fun" in the education domain because they reflect the same interest in Disney cartoon character "Mickey Mouse," presumably by a child in this family. Leveraging user behavior information from another domain will incorporate useful and noisy information at the same time. This raises another challenge, namely how to identify behavior from one domain that might be helpful to improve recommendations in the other domains while minimizing the impact of noisy information.

Recurrent Neural Networks (RNNs) have received a lot of attention in the context of SR [see, e.g., 19, 27, 35, 36]. However, most publications focus on a single domain and none simultaneously considers the shared account and cross-domain characteristics. In prior work that focuses on shared accounts, a common approach is to capture user preferences by extracting latent features from high-dimensional spaces that describe the relationships among users under the same account [2, 3, 12, 43, 45, 50]. And in prior work on the cross-domain task, one common solution is to aggregate information from two domains [13, 18, 21, 44], while another is to transfer knowledge from the source domain to target domain [12, 52]. None of these methods can be directly applied to SCSR for at least two reasons. Either important sequential characteristics of SR are largely ignored or they rely on explicit user ratings, which are usually unavailable in SR.

To address the above issues, we propose a novel **P**arallel **I**nformation-sharing **Net**work (π-Net) for SCSR. π-Net is organized in four main modules, namely a *sequence encoder*, a *shared account filter unit* (SFU), a *cross-domain transfer unit* (CTU) and a *hybrid recommendation decoder*. The *sequence encoder* module encodes the current user behavior sequence into a representation. Then, the SFU module identifies different user behaviors by learning user-specific representations and uses a gate mechanism to filter out information that might be useful for another domain. The output of the SFU is adopted to the CTU module, which transfers the information to another domain. The SFU and CTU are operated in a parallel recurrent way, which means that they can synchronously share information across both domains at each timestamp. Finally, the *hybrid recommendation decoder* module estimates the recommendation scores for each item based on the information from both domains. During learning, π-Net is jointly trained on two domains in an end-to-end fashion.

To assess the effectiveness of π-Net, we construct a new dataset, HVIDEO, which is defined based on real-world smart TV watching logs. To the best of our knowledge, HVIDEO is the first dataset that has shared account and cross-domain characteristics. We carry out extensive experiments on HVIDEO. The experimental results show that π-Net outperforms state-of-the-art baselines in terms of MRR and Recall.

Our contributions can be summarized as follows:

- We introduce the task of *Shared-account Cross-domain Sequential Recommendation* (SCSR), which has little attention in existing studies. We release a new shared account, smart TV recommendation dataset to facilitate future research in this space.
- We propose a novel π-Net model for SCSR, which simultaneously yields recommendations for two domains.
- We propose a *shared account filter unit* (SFU) and *cross-domain transfer unit* (CTU) that recurrently extract and share useful information between two domains to improve recommendation performances for both domains.

2 RELATED WORK

We consider three types of related work: sequential, shared account, and cross-domain recommendation.

2.1 Sequential recommendations

Traditional methods. The main traditional approach is based on Markov chains to predict users' next action given the last action [39]. Zimdars et al. [53] are the first to propose Markov chains for web page recommendation. They investigate how to extract sequential patterns to learn the next state using probabilistic decision-tree models. To further improve the performance, Shani et al. [39] propose a Markov Decision Process (MDP) based recommendation method, where the next recommendation can be computed through the transition probabilities among items. Yap et al. [48] introduce a competence score measure in personalized sequential pattern mining for next-item recommendations. Chen et al. [9] take playlists as Markov chains, and propose logistic Markov embeddings to learn the representations of songs for playlists prediction. A major issue of these methods is that the state space quickly becomes unmanageable when taking the whole sequence into account [27].

Deep learning based methods. Recently, RNNs have been devised to model variable-length sequential data. Quadrana et al. [35] have been the first to apply RNNs to sequential recommendation and achieve significant improvements over traditional methods. They utilize session-parallel mini-batch training and employ ranking-based loss functions to train the model. Later, they further propose data augmentation techniques to improve the performance of RNNs [42]. Bogina and Kuflik [6] explore user's dwell time based on an existing RNN-based framework by boosting items above a predefined dwell time threshold. Hidasi et al. [20] investigate how to add item property information such as text and images to an RNNs framework and introduce a number of parallel RNN (p-RNN) architectures; they propose alternative training strategies for p-RNNs that suit them better than standard training. Quadrana et al. [35] propose a hierarchical RNN model that can be used to generate personalized sequential recommendations. Li et al. [27] explore a hybrid encoder with an attention mechanism to model the user's sequential behavior and intent to capture the user's main purpose in the current sequence. Jiang et al. [24] propose an unsupervised learning-based model to identify users in the current sequence. They introduce an item-based sequence embedding technique by constructing a normalized random walk in the heterogeneous graph. Zhuang et al. [52] propose a novelty seeking model based on sequences in multi-domains to model an individual's propensity by transferring novelty seeking traits learned from a source domain for improving the accuracy of recommendations in the target domain.

Although there are many studies for sequential recommendations, none considers shared accounts and cross-domain simultaneously.

2.2 Shared account recommendations

Most previous approaches to shared account recommendations first identify users and then make personalized recommendations [2, 12, 43, 50]. Zhang et al. [49] are the first to study user identification, in which they use linear subspace clustering algorithms; they recommend the union of items that are most likely to be rated highly by each user. Bajaj and Shekhar [3] propose a similarity-based channel clustering method to group similar channels for IPTV accounts, and they use the Apriori algorithm to decompose users under an account. After that, they use personal profiles to recommend additional channels to the account. Wang et al. [45] suppose that different users consume services in different periods. They decompose users based on mining different preferences over different time periods from consumption logs. Finally, they use a standard User-KNN method to make recommendations for each identified user. Yang et al. [47] also analyze the similarity of the proportion of each type of items under a time period to judge whether a sequence is generated by the same user. Then they make recommendations to the specific user individually by recommending personalized genres to the identified users.

Yang et al. [46] identify users by using a projection based unsupervised method, and then use Factorization Machine techniques to predict a user's preference based on historical information to generate personalized recommendations. Jiang et al. [24] propose an unsupervised learning-based framework to differentiate the preferences of users and group sessions by users. They construct a heterogeneous graph to represent items and use a normalized random walk to create item-based session embeddings. A hybrid recommender is then proposed that linearly combines the account-level and user-level item recommendation by employing Bayesian personalized ranking matrix factorization (BPRMF) [37].

The differences between our method and above methods are at least two-fold. First, the work described above achieves user identification and recommendation in two separate processes, which means that the proposed models are not end-to-end learnable. Second, they do not consider the cross-domain scenario on which we focus.

2.3 Cross-domain recommendations

Cross-domain recommendation concerns data from multiple domains, which has proven to be helpful for alleviating the cold start problem [1, 5] and data sparsity issues [26, 33]. There is an assumption that there exists overlap in information between users and/or items across different domains [13, 14].

Traditional methods. There are two main ways in dealing with cross-domain recommendations [15]. One is to aggregate knowledge between two domains. Berkovsky et al. [4] propose four mediation techniques to solve the data sparsity problem by merging user preferences and extracting common attributes of users and items. Loni et al. [30] model user preference by using Matrix Factorization separately on different domains, and then incorporate user interaction patterns that are specific to particular types of items to generate recommendations on the target domain. Shapira et al. [40] compare several collaborative methods to demonstrate the effectiveness of utilizing available preference data from Facebook. Zhuang et al. [51] propose a consensus regularization classifier framework by considering both local data available in source domain and the prediction consensus with the classifiers learned from other source domains. Cao et al. [7] construct a nonparametric Bayesian framework by jointly considering multiple heterogeneous link prediction tasks between users and different types of items. Chen et al. [8] exploit the users and items shared between domains as a bridge to link different domains by embedding all users and items into a low-dimensional latent space between different domains.

The other approach to cross-domain recommendation is to transfer knowledge from source domain to target domain. Hu et al. [22] propose tensor-based factorization to share latent features between different domains. Cremonesi and Quadrana [12] propose a code-book-transfer to transfer rating patterns between domains. Kanagawa et al. [25] propose a content-based approach to learn the semantic information between domains. However, compared with deep learning methods, they are all shallow methods and have difficulties in learning complex user-item interactions.

Deep learning based methods. Deep learning is well suited to transfer learning as it can learn high-level abstractions among different domains. Lian et al. [28] first introduce a factorization framework to tie collaborative filtering and content-based filtering together; they use neural networks to build user and item embeddings. Elkahky et al. [13] propose a multi-view deep learning recommendation system by using rich auxiliary features to represent users from different domains based on *deep structured semantic model* (DSSM) [23]. Hu et al. [21] propose a model using a cross-stitch network [32] to learn complex user-item interaction relationships based on neural collaborative filtering [18]. Zhuang et al. [52] propose a graphic novelty-seeking model to fully characterize users' novelty-seeking trait so as to obtain better performances between different domains. Wang et al. [44] are the first to introduce the problem of cross-domain social recommendation, and they combine user-item interactions in information domains and user-user connections in social network services to recommend relevant items of information domains to target users of social domains; user and item attributes are leveraged to strengthen the embedding learning.

Although these studies have been proven effective in many applications, they are designed for static rating data, and cannot be applied to sequential recommendations, unlike the methods that we introduce in this paper.

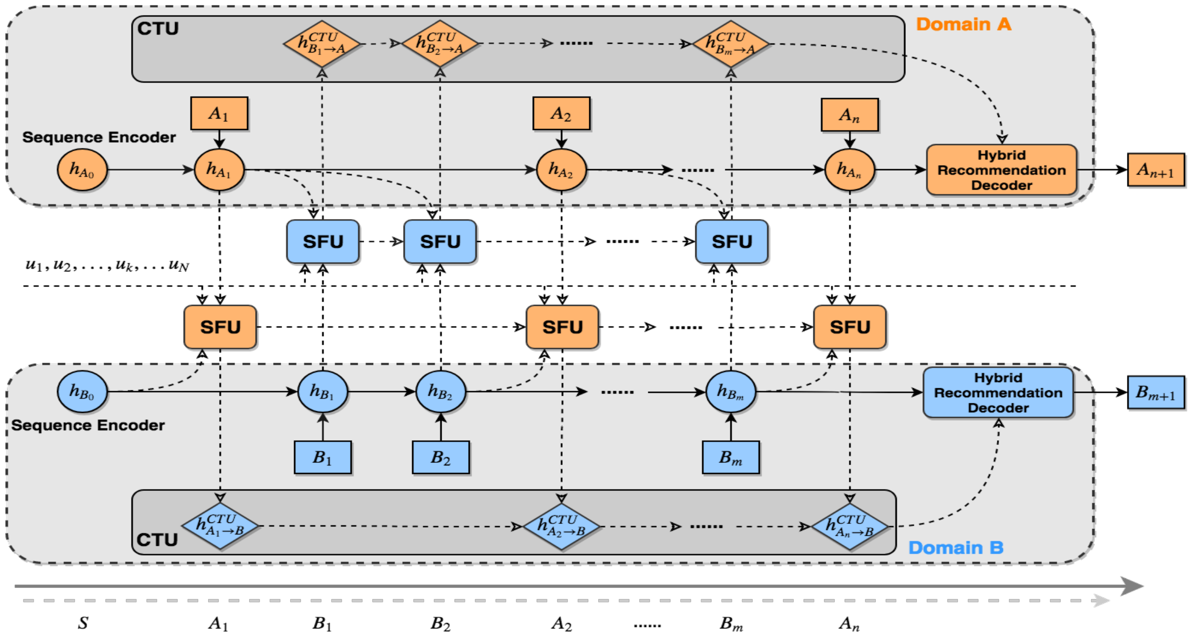

Figure 2: An overview of π-Net. Different colors represent different domains. Section 3.2 contains a walkthrough of the model.

3 METHOD

In this section, we first give a formulation of the SCSR problem. Then, we give a high-level introduction to the framework of π-Net. Finally, we describe each component of π-Net in detail.

3.1 Task definition

We represent a cross-domain behavior sequence (e.g., watching videos, listening to musics) from a shared account as $S = \{A_1, B_1, B_2, \ldots, A_i, \ldots, B_j, \ldots\}$, where $A_i \in \mathbb{A}$ $(1 \leq i \leq n)$ is the index of one consumed item in domain A; $\mathbb{A}$ is the set of all items in domain A; $B_j \in \mathbb{B}$ $(1 \leq j \leq m)$ is the index of one consumed item in domain B; $\mathbb{B}$ is the set of all items in domain B. Given S, SCSR tries to predict the next item by computing the recommendation probabilities for all candidates in two domains respectively, as shown in Eq. 1:

$$\begin{aligned} P(A_{i+1}|S) &\sim f_A(S) \\ P(B_{j+1}|S) &\sim f_B(S), \end{aligned} \tag{1}$$

where $P(A_{i+1}|S)$ denotes the probability of recommending the item A_{i+1} that will be consumed next in domain A. Also, $f_A(S)$ is the model or function to estimate $P(A_{i+1}|S)$. Similar definitions apply to $P(B_{j+1}|S)$ and $f_B(S)$.

The main differences between SCSR and traditional SR are two-fold. First, S is generated by multiple users (e.g., family members) in SCSR while it is usually generated by a single user in SR. Second, SCSR considers information from both domains for the particular recommendations in one domain, i.e., S is a mixture of behaviors from multiple domains. In this paper, we only consider two domains but the ideas easily generalize to multiple domains.

3.2 An overview of π-Net

In the following subsections, we will describe π-Net, our proposed solution for SCSR, in detail. The key idea of π-Net is to learn a recommendation model that can first extract some specific users' information from domain A, and then transfer the information to domain B, and combine it with the original information from domain B to improve recommendation performance for domain B, and vice versa. This process is achieved in a parallel way, which means that the information from both domains are shared recurrently at each timestamp.

Figure 2 provides an overview of π-Net. π-Net consists of four main components: a *sequence encoder*, a *shared account filter unit* (SFU), a *cross-domain transfer unit* (CTU) and a *hybrid recommendation decoder*. The *sequence encoder* encodes the behavior sequences of each domain into high-dimensional hidden representations. The shared account filter unit (SFU) takes each domain's representation as input and tries to extract the representation of specific users and ignore the others at each timestamp t. The cross-domain transfer unit (CTU) takes the information from the SFU as input, transforms it to another domain at each timestamp t, and combines the information from the two domains recurrently. The *hybrid recommendation decoder* integrates the information from both domains and generates a list of recommended items.

3.3 Sequence encoder

Like existing studies [19, 35, 42], we use a RNN to encode a sequence S. Here, we employ a gated recurrent unit (GRU) as the recurrent unit, with the GRU given as follows:

$$\begin{aligned} z_t &= \sigma(W_z[emb(x_t), h_{t-1}]) \\ r_t &= \sigma(W_r[emb(x_t), h_{t-1}]) \\ \widetilde{h_t} &= \tanh(W_{\widetilde{h}}[emb(x_t), r_t \odot h_{t-1}]) \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \widetilde{h_t}, \end{aligned} \tag{2}$$

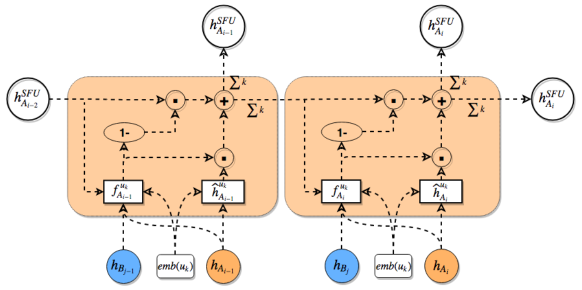

Figure 3: *Shared account filter unit* SFU. Domain A to domain B SFU.

where W_z, W_r, and $W_{\widetilde{h}}$ are weight matrices; $emb(x_t)$ is the item embedding of item x at timestamp t; σ denotes the sigmoid function. The initial state of the GRUs is set to zero vectors, i.e., $h_0 = 0$. Through the *sequence encoder* we obtain $H_A = \{h_{A_1}, h_{A_2}, \ldots, h_{A_i}, \ldots, h_{A_n}\}$ for domain A, and $H_B = \{h_{B_1}, h_{B_2}, \ldots, h_{B_j}, \ldots, h_{B_m}\}$ for domain B.

3.4 Shared-account filter unit

Under the shared account scenario, the behavior records under the same account are generated by different users. In practice, not all users that share the account are active in all domains. Besides, even though some users are active in the same domain, they may have different interests. For example, in the smart TV scenario, children may occupy the majority of the educational channel, while adults dominate the video TV channel.

The outputs H_A or H_B of the *sequence encoder* are the mixed representations of all users sharing the same account. To learn user-specific representations from the mixed representations, we propose a shared account filter unit (SFU) module, as shown in Figure 3. The SFU can be applied bidirectionally from "domain A to domain B" and "domain B to domain A". Here, we take the "domain A to domain B" direction and achieving recommendations in domain B as an example. To learn user-specific representations, we need to know the number of users for each account, which is, unfortunately, unavailable in most cases. In this work, we assume that there are K latent users (u_1, u_2, …, u_k, …, u_K) under each account. We first embed each latent user into $emb(u_k)$ ($1 \leq k \leq K$), which represents and encodes the latent interests of each user. Then, the specific representation for user u_k at timestamp i in domain A is defined in Eq. 3:

$$h^{u_k}_{A_i} = f^{u_k}_{A_i} \odot \widehat{h^{u_k}_{A_i}} + (1 - f^{u_k}_{A_i}) \odot h^{SFU}_{A_{i-1} \to B}. \tag{3}$$

Mathematically, the representation $h^{u_k}_{A_i}$ is a combination of two representations $h^{SFU}_{A_{i-1} \to B}$ and $\widehat{h^{u_k}_{A_i}}$ balanced by $f^{u_k}_{A_i}$. A higher value of $f^{u_k}_{A_i}$ means that item A_i is more likely generated by u_k and we should pay more attention to u_k's related representation $\widehat{h^{u_k}_{A_i}}$. A lower value of lower $f^{u_k}_{A_i}$ means that item A_i might not be related to u_k and we should inherit more information from previous time steps.

Next, we introduce the definitions of the three parts one by one.

(a) We propose a *distill gate* to implement $f^{u_k}_{A_i}$ in Eq. 4:

$$\begin{aligned} f^{u_k}_{A_i} = \sigma\Big(& W_{f_A} \cdot h_{A_i} + W_{f_B} \cdot h_{B_j} + \\ & U_f \cdot h^{SFU}_{A_{i-1} \to B} + V_f \cdot emb(u_k) + b_f\Big), \end{aligned} \tag{4}$$

where W_{f_A} , W_{f_B}, U_f and V_f are the parameters; b_f is the bias term; h_{A_i} and h_{B_j} are the mixed representations of domain A and B at timestamp i and j, respectively. $h^{SFU}_{A_{i-1} \to B}$ is the previous SFU output, which will be explained later. After the sigmoid function σ, each value of $f^{u_k}_{A_i}$ falls into $(0, 1)$. Thus, the *distill* score $f^{u_k}_{A_i}$ controls the amount of information of u_k to transfer from domain A to domain B. It should be noted that each latent representation $emb(uk)$ indicates the distribution of users with similar preference under one account, and it does not explicitly represents a specific user.

(b) $\widehat{h^{u_k}_{A_i}}$ is the candidate representation for u_k at timestamp i in domain A, which is computed based on the mixed representation h_{A_i}, the filtered previous SFU output $h^{SFU}_{A_{i-1} \to B}$, and the user u_k's own latent interest $emb(u_k)$, as shown in Eq. 5:

$$\begin{aligned} \widehat{h^{u_k}_{A_i}} = \tanh\Big(& W_h \cdot h_{A_i} + \\ & U_h \cdot h^{SFU}_{A_{i-1} \to B} + V_h \cdot emb(u_k) + b_h\Big), \end{aligned} \tag{5}$$

where W_h, U_h and V_h are the parameters; b_h is the bias term.

(c) $h^{SFU}_{A_i \to B}$ is the final output of the SFU at timestamp i in domain A, which is calculated as a combination of each user-specific representation $h^{u_k}_{A_i}$:

$$h^{SFU}_{A_i \to B} = \frac{1}{K} \sum_{k=1}^{K} \left(h^{u_k}_{A_i}\right). \tag{6}$$

Note that $h^{SFU}_{A_i \to B}$ is recurrently updated with Eq. 3 and 6.

3.5 Cross-domain transfer unit

The output of the SFU, $h^{SFU}_{A_i \to B}$, still belongs to the domain A. We need to transfer it to the domain B and then combine it with the original information in domain B to improve the recommendation performance for domain B. We propose the CTU to achieve this process. Specifically, we transform the representation $h^{SFU}_{A_i \to B}$ from domain A to domain B by employing another GRU to recurrently encode $h^{SFU}_{A_i \to B}$ at each time step to obtain a $h^{CTU}_{(A \to B)_i}$, as shown in Eq. 7:

$$h^{CTU}_{(A \to B)_i} = GRU(h^{SFU}_{A_i \to B}, h^{CTU}_{(A \to B)_{i-1}}), \tag{7}$$

where $h^{SFU}_{A_i \to B}$ is the representation filtered from domain A; $h^{CTU}_{(A \to B)_{i-1}}$ is the previous transformed representation in domain B. The initial state of the CTU is also set to zero vectors, i.e., $h^{CTU}_{(A \to B)_0} = 0$.

3.6 Hybrid recommendation decoder

The *hybrid recommendation decoder* integrates the hybrid information from both domains A and B to evaluate the recommendation probabilities of the candidate items. Specifically, it first gets the hybrid representation by concatenating the representation h_{B_j} from domain B and the transformed representation $h^{CTU}_{(A \to B)_i}$ from domain

A to domain B. Then, it evaluates the recommendation probability for each candidate item by calculating the matching of between the hybrid representation and the item-embedding matrix followed by a softmax function, as shown in Eq. 8:

$$P(B_{j+1}|S) = \text{softmax}\left(W_I \cdot \left[h_{B_j}, h^{CTU}_{(A\rightarrow B)_i}\right]^\top + b_I\right), \tag{8}$$

where W_I is the embedding matrix of all items of domain B; b_I is the bias term.

3.7 Objective function

We employ the negative log-likelihood loss function to train π-Net in each domain as follows:

$$\begin{aligned} L_A(\theta) &= -\frac{1}{|\mathbb{S}|}\sum_{S\in\mathbb{S}}\sum_{A_i\in S}\log P(A_{i+1}|S) \\ L_B(\theta) &= -\frac{1}{|\mathbb{S}|}\sum_{S\in\mathbb{S}}\sum_{B_j\in S}\log P(B_{j+1}|S), \end{aligned} \tag{9}$$

where θ are all the parameters of our model π-Net and $\mathbb{S}$ are the sequence instances in the training set. In the case of joint learning, the final loss is a linear combination of both losses:

$$L(\theta) = L_A(\theta) + L_B(\theta). \tag{10}$$

All parameters as well as the item embeddings are learned in an end-to-end back-propagation training way.

4 EXPERIMENTAL SETUP

4.1 Research questions

We aim to answer the following research questions:

(RQ1) What is the performance of π-Net in the SCSR task? Does it outperform the state-of-the-art methods in terms of Recall and MRR? (See Section 5.)

(RQ2) What are the performances of π-Net on different domains? Does it improve the performance of both domains? Are the improvements equivalent? (See Section 5.)

(RQ3) Is it helpful to model the shared-account characteristic? How well does SFU improve the performance of recommendations? (See Section 6.1.)

(RQ4) Is it helpful to leverage the cross-domain information? How well does CTU improve the performance of recommendations? (See Section 6.1.)

(RQ5) How does the hyperparameter K (the number of latent users) affect the performance of π-Net? Does the best K accord with reality? (See Section 6.2.)

4.2 Dataset

There is no publicly available dataset for SCSR. To demonstrate the effectiveness of the proposed π-Net model, we build and release a new dataset, named HVIDEO. HVIDEO is a smart TV dataset that contains 260k users watching logs from October 1st 2016 to June 30th 2017. The logs are collected on two platforms (the V-domain and the E-domain) from a well-known smart TV service provider. The V-domain contains family *video* watching behavior including TV series, movies, cartoons, talent shows and other programs. The E-domain covers online *educational* videos based on textbooks from elementary to high school, as well as instructional videos on sports, food, medical, etc. On the two platforms, we gather user behaviors, including which video is played, when a smart TV starts to play a video, and when it stops playing the video, and how long the video has been watched. Compared with previous datasets, HVIDEO contains rich and natural family behavior data generated in shared-account and cross-domain context. Therefore, it is very suitable for SCSR research.

Table 1: Statistics of the HVIDEO dataset.

V-domain	
#Items	16,407
#Logs	227,390
E-domain	
#Items	3,380
#Logs	177,758
#Overlapped-users	13,714
#Sequences	134,349
#Training-sequences	102,182
#Test-sequences	13,201
#Validation-sequences	18,966

We conduct some preprocessing on the dataset. We first filter out users who have less than 10 watching records and whose watching time is less than 300 seconds. Then, we merge records of the same item watched by the same user with an adjacent time less than 10 minutes. After that, we combine data from different domains together in chronological order which is grouped by the same account.

Because each account has a lot of logs recorded in a long time, we split the logs from each account into several small sequences with each containing 30 watching records. And we require that the number of items in both domains must be greater than 5 in each sequence, which can ensure the sequences mix is high enough.

For evaluation, we use the last watched item in each sequence for each domain as the ground truth respectively. We randomly select 75% of all data as the training set, 15% as the validation set, and the remaining 10% as the test set. The statistics of the final dataset are shown in Table 1.

4.3 Baseline methods

For our contrastive experiments, we consider baselines from four categories: traditional, sequential, shared account, and cross-domain recommendations.

4.3.1 Traditional recommendations. As traditional recommendation methods, we consider the following:

- POP: Ranks items in the training set based on their popularity, and always recommends the most popular items. Frequently used as a baseline because of its simplicity [18].
- Item-KNN: Computes an item-to-item similarity that is defined as the number of co-occurrences of two items within sequences divided by the square root of the product of the number of sequences in which either item occurs. Regularization is included to avoid coincidental high similarities between rarely visited items [29].
- BPR-MF: A commonly used matrix factorization method. Like [19], we apply it for sequential recommendations by representing

a new sequence with the average latent factors of items that appeared in this sequence so far.

4.3.2 Shared account recommendations. There are some studies that explore shared account recommendations by first achieving user identification [3, 24, 47]. However, they need extra information for user identification, e.g., some explicit ratings for movies or descriptions for some musics, even some textual descriptions for flight tickets, which is not available in HVIDEO. Therefore, we select a method that works on the IP-TV recommendation task that is similar to ours.

- VUI: Encompasses an algorithm to decompose members in a composite account by mining different preferences over different time periods from logs [45]. The method first divides a day into time periods, so the logs of each account fall into the corresponding time period; logs in each time period are assumed to be generated by a virtual user that is represented by a 3-dimensional $\{account \times item \times time\}$ vector. After that, cosine similarity is used to calculate similarity among virtual users, some of which are merged into a latent user. VUI deploys User-KNN method to produce recommendations for these latent users.

4.3.3 Cross-domain recommendations. As cross-domain recommendations, we choose two baseline methods.

- NCF-MLP++: Uses a deep learning-based process to learn the inner product of the traditional collaborative filtering by using Multilayer perceptron (MLP) [18]. We improve NCF-MLP by sharing the collaborative filtering in different domains. It is too time-consuming to rank all items with this method, because it needs to compute the score for each item one by one. We randomly sample four negative instances for each positive instance in the training process, and sample 3,000 negatives for each predicted target item in the test process, thus simplifying the task for this method.
- Conet: A neural transfer model across domains on the basis of a cross-stitch network [21, 32], where a neural collaborative filtering model [18] is employed to share information between domains.

4.3.4 Sequential recommendations. As sequential recommendations methods we consider two methods with somewhat similar architectures as π-Net.

- GRU4REC: Uses GRU to encode sequential information and employs ranking-based loss functions for learning the model [19].
- HGRU4REC: Takes the user's information into account, and proposes a hierarchical RNN model based on GRU4REC [35]. It incorporates auxiliary features into GRU networks to improve the sequential recommendations.

4.4 Evaluation metrics

Recommender systems can only recommend a limited number of items at a time. The item a user might pick should be amongst the first few on the ranked list [10, 17, 35]. Commonly used metrics are MRR@20 and Recall@20 [27, 31, 36]. In this paper, we also report MRR@5, Recall@5 and MRR@10, Recall@10.

- Recall: The primary evaluation metric is Recall, which measures the proportion of cases when the relevant item is amongst the top ranked items in all test cases.
- MRR: Another used metric is MRR (Mean Reciprocal Rank), which is the average of reciprocal ranks of the relevant items. And the reciprocal rank is set to zero if the ground truth item is not in the recommendation list. MRR takes the rank of the items into consideration, which is vital in settings where the order of recommendations matters. We choose MRR instead of other ranking metrics, because there is only one ground truth item for each recommendation; no ratings or grade levels are available.

4.5 Implementation details

We set the item embedding size and GRU hidden state size to 90. We use dropout [41] with drop ratio $p = 0.8$. These settings are chosen with grid search on the validation set. For the latent user size K in Section 3.4, we try different settings, the analysis of which can be found in Section 6.2. We initialize model parameters randomly using the Xavier method [16]. We take Adam as our optimizing algorithm. For the hyper-parameters of the Adam optimizer, we set the learning rate $\alpha = 0.001$. We also apply gradient clipping [34] with range $[-5, 5]$ during training. To speed up the training and converge quickly, we use mini-batch size 64. We test the model performance on the validation set for every epoch. π-Net is implemented in Tensorflow and trained on a GeForce GTX TitanX GPU. The code used to run the experiments and HVIDEO dataset released in this paper is available online.[1]

5 EXPERIMENTAL RESULTS (RQ1 & RQ2)

The results of π-Net and the baseline methods are shown in Table 2. We can see that π-Net outperforms all baselines in terms of MRR and Recall for all reported values. We can gain several insights from Table 2.

First, π-Net significantly outperforms all baselines and achieves the best results on all metrics. Some strong baselines, i.e., GRU4REC and HGRU4REC, are also outperformed by $\pi-$Net. The improvements indicate that considering the specific shared account and cross-domain setting is helpful for sequential recommendations. The increase over GRU4REC is 13.03% (at most) in terms of Recall, and 3.18% in terms of MRR. And the increase over HGRU4REC is 7.67% (at most) in terms of Recall, and 0.77% in terms of MRR. We believe that those performance improvements are due to the fact that π-Net contains two main components for as part of its end-to-end recommendation model, namely the SFU and CTU. With these two modules, π-Net is able to model user preferences more accurately by leveraging complementary information from both domains and improve recommendation performance in both domains. As an aside, GRU4REC and HGRU4REC have very similar architectures as π-Net and we re-implement them within the same TensorFlow framework, so as to obtain a fair comparison. We will analyze the effects of the SFU and CTU in more depth in Section 6.1.

Second, we can observe that the Recall values of π-Net on the V-domain are better than those on the E-domain. This is also true for the other methods. Most accounts have much more data on the V-domain due to its content diversity; because of this, models can better learn users viewing characteristics on the V-domain. Additionally, comparing π-Net with the best baseline, HGRU4REC, we find that the largest increase on the E-domain is larger than on

[1] https://bitbucket.org/Catherine_Ma/sigir2019_muyang_recommendation/

Table 2: Experimental results (%) on the HVIDEO dataset.

Categories	Methods	V-domain recommendation						E-domain recommendation					
		MRR			Recall			MRR			Recall		
		@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
Traditional	POP	2.66	3.07	3.27	5.01	7.77	10.49	1.71	1.96	2.24	2.21	3.61	6.58
	Item-KNN	4.43	4.16	2.93	10.48	16.49	23.93	2.11	2.39	2.90	3.01	5.77	12.11
	BPR-MF	1.21	1.31	1.36	1.88	2.56	3.38	1.34	1.52	1.64	2.74	4.05	5.83
Shared account	VUI-KNN	3.44	3.53	2.87	16.46	24.85	34.76	2.03	2.51	3.48	6.36	11.55	24.27
Cross domain	NCF-MLP++	16.25	17.25	17.90	26.10	33.61	43.04	3.92	4.57	5.14	7.36	12.27	20.84
	Conet	21.25	22.61	23.28	32.94	43.07	52.72	5.01	5.63	6.21	9.26	14.07	22.71
SR	GRU4REC	78.27	78.46	78.27	80.15	81.63	83.04	12.27	13.00	13.70	16.24	21.89	32.16
	HGRU4REC	80.37	80.53	80.62	81.92	83.21	84.43	14.47	15.37	16.11	19.79	26.72	37.52
SCSR	π-Net	**80.51**	**80.80**	**80.95**	**83.22**†	**85.34**†	**87.48**†	**14.63**	**15.83**	**16.88**†	**20.41**†	**29.61**†	**45.19**†

Bold face indicates the best result in terms of the corresponding metric. Significant improvements over the best baseline results are marked with † (t-test, $p < .05$). To ensure a fair comparison, we re-implemented GRUE4REC and HGRU4REC in Tensorflow, just like π-Net; the final results may be slightly different from the Theano version released by the authors. To obtain the results of NCF-MLP++ and Conet, we run the code released by Hu et al. [21].

V-domain. The largest increase in Recall is 3.05% on the V-domain and 7.67% on the E-domain. And for the MRR values, the largest increase is 0.33% on the V-domain, and 0.77% on the E-domain. As before, this difference is relative gain is because the V-domain contains much more data than the E-domain. Therefore, the space for potential improvements on the V-domain is smaller than that on the E-domain after considering shared account and cross-domain information.

Third, RNN-based methods perform better than traditional methods, which demonstrates that RNN-based methods are good at dealing with sequential information. They are able to learn better dense representations of the data through nonlinear modeling, which we assume is able to provide a higher level of abstraction. The shared account and cross-domain baselines (e.g., VUI-KNN, NCF-MLP++ and Conet) perform much worse than π-Net. They also perform substantially worse than HGRU4REC. This is because these shared account and cross-domain baselines ignore sequential information, VUI-KNN only considers the length of watching time, and NCF-MLP++ and Conet do not use any time information. Another reason is that NCF-MLP++ and Conet just map each overlapped account in both domains to the same latent space to calculate the user-item similarity. However, the existence of shared accounts makes it difficult to find the same latent space for different latent users under one account. Besides, VUI-KNN is not a deep learning method and it simply distinguishes users based on the fixed divided time periods, which means it assumes there is only one member in each time period. However, in the smart TV scenario, many people usually watch programs together [45]. This situation cannot be captured very well by VUI-KNN. In contrast, the SFU can extract components of each hidden user according to their viewing records in another domain, which proves to be informative. We can also see the results of BPR-MF are lower than POP, which indicates that most items users watched are very popular, which is also in line with the phenomenon of people like pursuing popularity.

Fourth, the increase in Recall is greater than the increase in MRR. As for π-Net, Recall increases by 24.78% from Recall@5 to Recall@20 while MRR only increases by 2.25% from MRR@5 to MRR@20. This is because Recall measures the proportion of relevant items when they are amongst the top-k list, while MRR takes the rank of the relevant items into consideration. As the size of k increases, the number of relevant items will increase, and consequently, Recall values will increase. However, the calculation of MRR is the reciprocal of the ranking of each positive item. So an increase k is bound to have a limited impact on the MRR.

6 EXPERIMENTAL ANALYSIS

6.1 Analysis of SFU and CTU (RQ3 & RQ4)

We design experiments to verify the effectiveness of the proposed SFU and CTU. The results are listed in Table 3. There, we compare π-Net with π-Net (no SFU, no CTU) and π-Net (no SFU). π-Net (no SFU, no CTU) removes both SFU and CTU, which is actually GRU4REC except that π-Net (no SFU, no CTU) is jointly trained with a different loss. π-Net (no SFU) is π-Net without the shared-account filter part, which means it makes recommendations without considering filtering the information of each latent user. To achieve this, π-Net (no SFU) uses the output of the *sequence encoder* as the input of the CTU part.

From Table 3, we can see that π-Net (no SFU) performs better than π-Net (no SFU, no CTU). This is because π-Net (no SFU) leverages information from both domains when making next-item prediction. On V-domain recommendation, the largest increase is 0.69% in terms of MRR@20, and 2.4% in terms of Recall@20. On E-domain recommendation, π-Net (no SFU) improves by 3.66% in terms of MRR@5, and 4.43% in terms of Recall@5. This confirms that CTU is able to improve the recommendation performance by transferring information recurrently.

From Table 3, we can also observe that π-Net outperforms π-Net (no SFU) in terms of most metrics, which means extracting information of latent users under the same account to another domain can greatly improve recommendation performances in both domains. On V-domain recommendation, MRR values can increase by 1.98%

Table 3: Analysis of the SFU and CTU.

Methods	V-domain recommendation						E-domain recommendation					
	MRR			Recall			MRR			Recall		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
π-Net (no SFU, no CTU)	78.02	78.17	78.28	80.13	81.34	82.93	12.69	13.43	14.05	16.54	22.29	31.45
π-Net (no SFU)	78.59	78.85	78.97	81.71	83.58	85.33	**16.35**	**17.04**	**17.59**	**20.97**	26.29	34.44
π-Net	**80.51**†	**80.80**†	**80.95**†	**83.22**†	**85.34**†	**87.48**†	14.63	15.83	16.88	20.41	**29.61**†	**45.19**†

π-Net (no SFU) is π-Net without SFU. π-Net (no SFU, no CTU) is π-Net without SFU and CTU. SFU relies on CTU and SFU will not exist without CTU, so there is no π-Net (no CTU). Significant improvements over π-Net (no SFU) are marked with † (t-test, $p < .05$).

Table 4: Analysis of the hyperparameter K.

K values	V-domain recommendation						E-domain recommendation					
	MRR			Recall			MRR			Recall		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
1	80.19	80.50	80.66	82.85	85.15	87.40	13.92	15.06	16.10	19.76	28.74	43.98
2	80.48	80.75	80.91	83.08	85.06	87.31	14.29	15.47	16.54	19.83	28.96	44.77
3	80.53	80.79	80.93	**83.34**	85.31	87.31	14.45	15.54	16.64	20.23	28.61	44.64
4	80.51	80.80	80.95	83.22	**85.34**	**87.48**	**14.63**	**15.83**	**16.88**	20.41	**29.61**	**45.19**
5	**80.60**	**80.86**	**81.02**	83.25	85.19	87.47	14.59	15.71	16.75	**20.42**	28.97	44.38

at most and 1.92% at least, Recall values can increase by 2.15% at most and 1.51% at least. On E-domain recommendation, Recall@20 increases by 10.75%. But some metrics decrease a little; we believe this is a sign of noise brought in through cross-domain information. The proposed SFU sometimes fails to distinguish between noisy and valuable information; it remains as a topic for future work to design even better SFU modules.

We find that the gap between π-Net (no SFU) and π-Net (no SFU, no CTU) is larger than that between π-Net and π-Net (no SFU). π-Net (no SFU) is even better than π-Net on E-domain in terms of MRR. This indicates that the CTU is more effective than the SFU in π-Net. Almost all members have viewing records in the V-domain. However, items on the E-domain are mostly educational programs, so children take up a large proportion, and their educational interests are relatively fixed. As a result, the information extracted by the SFU from the V-domain mostly belongs to children, which is less helpful because we already have enough data on the E-domain to learn such features in most cases.

Additionally, we see that π-Net (no SFU, no CTU) gets the lowest performance amongst the three these methods, while it still outperforms most of the baselines listed in Table 2. Meanwhile, π-Net (no SFU) outperforms all sequential recommendation baselines. In summary, then, combining information from an auxiliary domain is useful.

6.2 Influence of hyperparameter K (RQ5)

Family members share a single account in the HVIDEO dataset. We have proposed the SFU to model this, which introduces a hyperparameter K, representing the number of latent users. To study how the setting K affects recommendation performance, we compare different values of K while keeping other settings unchanged. Taking into account the popular sizes of families, we consider $K = 1, \ldots, 5$.

As shown in Table 4, we can see that the best values of MRR and Recall are achieved when $K = 3, 4, 5$, and especially $K = 4$. This is mostly consistent with the size of modern families. The lowest MRR and Recall values are achieved when $K = 1$. Although K can affect the recommendation performance, the influence is small. The largest gap between the best and worst performances is only 1.21% in terms of Recall and 0.78% in terms of MRR. Even if $K = 1$, our model still considers the information of all members except that all members are modeled as a single latent user.

7 CONCLUSION AND FUTURE WORK

In this paper, we have proposed the new task of SCSR. To address this task, we have proposed a novel parallel information-sharing network named π-Net. By leveraging a shared account filter and a cross-domain transfer mechanism, π-Net is able to improve recommendations performance. To show the effectiveness of π-Net, we have conducted experiments on a newly created smart TV dataset, that will be released upon publication of the paper. Experimental results demonstrate that π-Net outperforms state-of-the-art methods in terms of MRR and Recall.

A limitation of π-Net is that it works better only when we have shared information in two domains that are complementary to each other. When there is only one domain or the data in two domains share less information, π-Net is not as effective. As to future work, π-Net can be advanced in two directions. First, better SFU modules can be designed. Especially, we assume the same latent users for all shared accounts in this study. This can be further improved by automatically detecting the number of family members. Second, to avoid the influence of other factors, we have simplified the architecture of π-Net (e.g., encoders, decoders and loss functions). It would be interesting to see whether more complex architectures will further improve the performance of π-Net.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable comments. This work is supported by the Natural Science Foundation of China (61672324, 61672322), the Natural Science Foundation of Shandong province (2016ZRE27468), the Tencent AI Lab Rhino-Bird Focused Research Program (JR201932), the Fundamental Research Funds of Shandong University, Ahold Delhaize, the Association of Universities in the Netherlands (VSNU), and the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Fabian Abel, Eelco Herder, Geert-Jan Houben, Nicola Henze, and Daniel Krause. 2013. Cross-system user modeling and personalization on the social web. *User Modeling and User-Adapted Interaction* 23 (2013), 169–209.

[2] Michal Aharon, Eshcar Hillel, Amit Kagian, Ronny Lempel, Hayim Makabee, and Raz Nissim. 2015. Watch-it-next: a contextual TV recommendation system. In *ECML-PKDD 2015*. 180–195.

[3] Payal Bajaj and Sumit Shekhar. 2016. Experience individualization on online TV platforms through persona-based account decomposition. In *MM 2016*. 252–256.

[4] Shlomo Berkovsky, Tsvi Kuflik, and Francesco Ricci. 2007. Cross-domain mediation in collaborative filtering. In *UMAP 2007*. 355–359.

[5] Shlomo Berkovsky, Tsvi Kuflik, and Francesco Ricci. 2008. Mediation of user models for enhanced personalization in recommender systems. *User Modeling and User-Adapted Interaction* 18 (2008), 245–286.

[6] Veronika Bogina and Tsvi Kuflik. 2017. Incorporating dwell time in session-based recommendations with recurrent Neural networks. In *RecSys 2017*. 57–59.

[7] Bin Cao, Nathan N Liu, and Qiang Yang. 2010. Transfer learning for collective link prediction in multiple heterogeneous domains. In *ICML 2010*. 159–166.

[8] Leihui Chen, Jianbing Zheng, Ming Gao, Aoying Zhou, Wei Zeng, and Hui Chen. 2017. TLRec: transfer learning for cross-domain recommendation. In *ICBK 2017*. 167–172.

[9] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. 2012. Playlist prediction via metric embedding. In *SIGKDD 2012*. 714–722.

[10] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan S. Kankanhalli. 2018. Aspect-aware latent factor model: rating prediction with ratings and reviews. In *WWW 2018*. 639–648.

[11] Zhiyong Cheng, Jialie Shen, Lei Zhu, Mohan S Kankanhalli, and Liqiang Nie. 2017. Exploiting Music Play Sequence for Music Recommendation. In *IJCAI*. 3654–3660.

[12] Paolo Cremonesi and Massimo Quadrana. 2014. Cross-domain recommendations without overlapping data: myth or reality?. In *RecSys 2014*. 297–300.

[13] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *WWW 2015*. 278–288.

[14] Aleksandr Farseev, Ivan Samborskii, Andrey Filchenkov, and Tat-Seng Chua. 2017. Cross-domain recommendation via clustering on multi-layer graphs. In *SIGIR 2017*. 195–204.

[15] Ignacio Fernández-Tobías, Iván Cantador, Marius Kaminskas, and Francesco Ricci. 2012. Cross-domain recommender systems: a survey of the state of the art. In *CERI 2012*. 24.

[16] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS 2010*. 249–256.

[17] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial personalized ranking for recommendation. In *SIGIR 2018*. 355–364.

[18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW 2017*. 173–182.

[19] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *ICLR 2016*.

[20] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *RecSys 2016*. 241–248.

[21] Guangneng Hu, Yu Zhang, and Qiang Yang. 2018. CoNet: collaborative cross networks for cross-domain recommendation. In *CIKM 2018*. 667–676.

[22] Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Can Zhu. 2013. Personalized recommendation via cross-domain triadic factorization. In *WWW 2013*. 595–606.

[23] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *CIKM 2013*. 2333–2338.

[24] Jyun-Yu Jiang, Cheng-Te Li, Yian Chen, and Wei Wang. 2018. Identifying users behind shared accounts in online streaming services. In *SIGIR 2018*. 65–74.

[25] Heishiro Kanagawa, Hayato Kobayashi, Nobuyuki Shimizu, Yukihiro Tagami, and Taiji Suzuki. 2018. Cross-domain recommendation via deep domain adaptation. *arXiv preprint arXiv:1803.03018* abs/1803.03018 (2018).

[26] Bin Li, Qiang Yang, and Xiangyang Xue. 2009. Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In *IJCAI 2009*. 2052–2057.

[27] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *CIKM 2017*. 1419–1428.

[28] Jianxun Lian, Fuzheng Zhang, Xing Xie, and Guangzhong Sun. 2017. CCCFNet: a content-boosted collaborative filtering neural network for cross domain recommender systems. In *WWW 2017*. 817–818.

[29] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing* 1 (2003), 76–80.

[30] Babak Loni, Yue Shi, Martha Larson, and Alan Hanjalic. 2014. Cross-domain collaborative filtering with factorization machines. In *CERI 2014*. 656–661.

[31] Lei Mei, Pengjie Ren, Zhumin Chen, Liqiang Nie, Jun Ma, and Jian-Yun Nie. 2018. An attentive interaction network for context-aware recommendations. In *CIKM 2018*. 157–166.

[32] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch networks for multi-task learning. In *CVPR 2016*. 3994–4003.

[33] Weike Pan, Evan Wei Xiang, Nathan Nan Liu, and Qiang Yang. 2010. Transfer learning in collaborative filtering for sparsity reduction. In *AAAI 2010*. 230–235.

[34] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *ICML 2013*. 1310–1318.

[35] Massimo Quadrana, Alexandros Karatzoglou, Balzs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *RecSys 2017*. 130–137.

[36] Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2019. RepeatNet: a repeat aware neural recommendation machine for session-based recommendation. In *AAAI 2019*.

[37] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: bayesian personalized ranking from implicit feedback. In *UAI 2009*. 452–461.

[38] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW 2001*. 285–295.

[39] Guy Shani, David Heckerman, and Ronen I. Brafman. 2005. An mdp-based recommender system. *Journal of Machine Learning Research* 6 (2005), 1265–1295.

[40] Bracha Shapira, Lior Rokach, and Shirley Freilikhman. 2013. Facebook single and cross domain data for recommendation systems. *User Modeling and User-Adapted Interaction* 23 (2013), 211–247.

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15 (2014), 1929–1958.

[42] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *RecSys 2016*. 17–22.

[43] Koen Verstrepen and Bart Goethals. 2015. Top-n recommendation for shared accounts. In *RecSys 2015*. 59–66.

[44] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item silk road: recommending items from information domains to social users. In *SIGIR 2017*. 185–194.

[45] Zhijin Wang, Yan Yang, Liang He, and Junzhong Gu. 2014. User identification within a shared account: improving IP-TV recommender performance. In *ADBIS 2014*. 219–233.

[46] Shuo Yang, Somdeb Sarkhel, Saayan Mitra, and Viswanathan Swaminathan. 2017. Personalized video recommendations for shared accounts. In *ISM 2017*. 256–259.

[47] Yan Yang, Qinmin Hu, Liang He, Minjie Ni, and Zhijin Wang. 2015. Adaptive temporal model for IPTV recommendation. In *ICWAIM 2015*. 260–271.

[48] Ghim-Eng Yap, Xiao-Li Li, and S Yu Philip. 2012. Effective next-items recommendation via personalized sequential pattern mining. In *DASFAA 2012*. 48–64.

[49] Amy Zhang, Nadia Fawaz, Stratis Ioannidis, and Andrea Montanari. 2012. Guess who rated this movie: identifying users through subspace clustering. In *UAI 2012*. 944–953.

[50] Yafeng Zhao, Jian Cao, and Yudong Tan. 2016. Passenger prediction in shared accounts for flight service recommendation. In *APSCC 2016*. 159–172.

[51] Fuzhen Zhuang, Ping Luo, Hui Xiong, Yuhong Xiong, Qing He, and Zhongzhi Shi. 2010. Cross-domain learning from multiple sources: a consensus regularization perspective. *Transactions on Knowledge and Data Engineering* 22 (2010), 1664–1678.

[52] Fuzhen Zhuang, Yingmin Zhou, Fuzheng Zhang, Xiang Ao, Xing Xie, and Qing He. 2018. Cross-domain novelty seeking trait mining for sequential recommendation. *arXiv preprint arXiv:1803.01542* (2018).

[53] Andrew Zimdars, David Maxwell Chickering, and Christopher Meek. 2001. Using temporal data for making recommendations. In *UAI 2001*. 580–588.