Item Silk Road: Recommending Items from Information Domains to Social Users

Xiang Wang
National University of Singapore
xiangwang@u.nus.edu

Xiangnan He*
National University of Singapore
xiangnanhe@gmail.com

Liqiang Nie
ShanDong University
nieliqiang@gmail.com

Tat-Seng Chua
National University of Singapore
dcscts@nus.edu.sg

ABSTRACT

Online platforms can be divided into information-oriented and social-oriented domains. The former refers to forums or E-commerce sites that emphasize user-item interactions, like Trip.com and Amazon; whereas the latter refers to social networking services (SNSs) that have rich user-user connections, such as Facebook and Twitter. Despite their heterogeneity, these two domains can be bridged by a few overlapping users, dubbed as *bridge users*. In this work, we address the problem of *cross-domain social recommendation*, *i.e.*, recommending relevant items of information domains to potential users of social networks. To our knowledge, this is a new problem that has rarely been studied before.

Existing cross-domain recommender systems are unsuitable for this task since they have either focused on homogeneous information domains or assumed that users are fully overlapped. Towards this end, we present a novel *Neural Social Collaborative Ranking* (NSCR) approach, which seamlessly sews up the user-item interactions in information domains and user-user connections in SNSs. In the information domain part, the attributes of users and items are leveraged to strengthen the embedding learning of users and items. In the SNS part, the embeddings of bridge users are propagated to learn the embeddings of other non-bridge users. Extensive experiments on two real-world datasets demonstrate the effectiveness and rationality of our NSCR method.

CCS CONCEPTS

•**Information systems → Social recommendation; Retrieval models and ranking; Recommender systems;**

KEYWORDS

Cross-domain Recommendation, Deep Collaborative Filtering, Neural Network, Deep Learning

---

*Xiangnan He is the corresponding author.

---

1 INTRODUCTION

Nowadays online platforms play a pivotal role in our daily life and encourage people to share experiences, exchange thoughts, and enjoy online services. Regardless of applications, we can roughly divide the existing platforms into information-oriented and social-oriented domains. The former typically refers to forums or E-Commerce sites that have thorough knowledge on items, such as point-of-interests in Trip.com, movies in IMDb, and products in Amazon. These sites have ample user-item interactions available in the form of users' reviews, ratings, along with various kinds of implicit feedback like views and clicks [1]. On the other hand, the social-oriented domains are mainly social network sites, which emphasize the social connections among users [15].

When adopting an item, besides consulting the information sites, a user usually gathers more detailed information from her experienced friends. This refers to *word-of-mouth marketing*, which is widely recognized as the most effective strategy for producing recommendation. As reported by Cognizant[1], more than 45% of travelers rely on social networks to seek advice from friends for travel. However, most existing SNSs, like Facebook and Twitter, are designed mainly for users to rebuild their real-world connections, rather than for seeking options regarding items. Though some item cues implying users' preference can be found in SNSs, they typically contain item names only with limited details. The sparse and weak user-item interactions greatly hinder the ability of SNSs to offer item recommendation services.

Fortunately, some users may be simultaneously involved in both SNSs and information-domain sites, who can act as a bridge to propagate user-item interactions across domains. For example, it is not unusual for a user to share her travel experiences in Trip.com; and if the user also holds a Facebook account, we can recommend her friends in Facebook with her liked items from Trip.com. In social circles, these bridge users are like the *silk road* to route relevant items from information domains to (non-bridge) users of social networks. As such, we formulate the task of *cross-domain social recommendation*, which aims to recommend relevant items of information domains to the users of social domains. Apparently, this task is related to the recently emerging topic — cross-domain recommendation [13]. However, we argue that existing efforts have either focused on homogeneous domains (*i.e.,* multiple sites of the information domain) [5], or unrealistically assumed that the users are fully overlapped [13, 30]. Our task to address is particularly challenging due to the following two practical considerations.

---

[1]https://www.cognizant.com.

- Insufficient bridge users. To gain a deep insight, we analyzed the overlapped users between Trip.com and Facebook/Twitter, finding that only 10.5% of 8, 196 Facebook users and 6.9% of 7, 233 Twitter users have public accounts in Trip.com. It is highly challenging to leverage history of such limited number of bridge users to provide quality recommendation for non-bridge users.
- Rich attributes. The users and items of an information domain are usually associated with rich attributes. For instance, Trip.com enables users to indicate their travel preference explicitly, and associates travel spots (*i.e.,* items) with specific travel modes, among other information. However, little attention has been paid to leverage these attributes to boost the performance of cross-domain recommendation.

In this work, we propose a novel solution named *Neural Social Collaborative Ranking* (NSCR) for the new task of cross-domain social recommendation. It is developed based on the recent advance of neural collaborative filtering (NCF) [11], which is further extended to model the cross-domain social relations by combining with the graph regularization technique [9]. We entail two key technical components of our NSCR as follows.

- For the modelling of information domain, we build an attribute-aware recommender based on the NCF framework. To fully exploit the interactions among a user, an item, and their attributes, we enhance NCF by plugging a *pairwise pooling* operation above the embedding vectors of user (item) ID and attributes. In contrast to the default average pooling used by NCF [11] and other recent neural recommenders [4], our use of pairwise pooling better captures feature interactions in the low level [10, 21], greatly facilitating the following deep layers to learn higher-order interactions among users, items and attributes.
- For the modelling of social domain, it is natural to guide the embedding learning of social users by using the embeddings of bridge users. As the embeddings of bridge users are optimized to predict user–item interactions (*e.g.,* ratings and purchases), propagating their embeddings to social users helps to bridge the heterogeneity gap between information domain and social domain. To implement such propagation effect, we employ the *smoothness* constraint (*i.e.,* graph Laplacian) on the social network, which enforces close friends to have similar embedding so as to reflect their similar preferences.

To sum up, the key contributions of this work are three-fold:

(1) To our knowledge, we are the first to introduce the task of cross-domain social recommendation, which recommends relevant items of information domains to target users of social domains.
(2) We propose a novel solution that unifies the strengths of deep neural networks in modelling attributed user-item interactions and graph Laplacian in modelling user-user social relations.
(3) We construct two real-world benchmark datasets for exploring the new task of cross-domain social recommendation and extensively evaluate our proposed solution.

2 PRELIMINARY

We first formulate the task of cross-domain social recommendation, and then shortly recapitulate the matrix factorization model, highlighting its limitations for addressing the task.
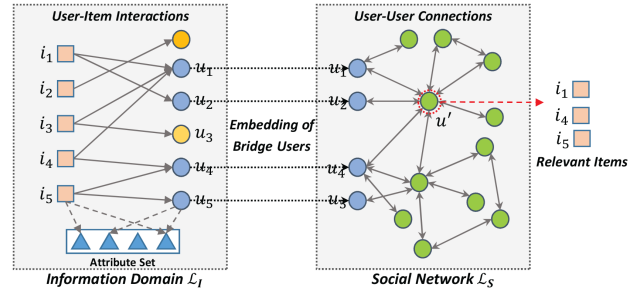

Figure 1: Illustration of the cross-domain social recommendation task.

2.1 Problem Formulation

Figure 1 illustrates the task of cross-domain social recommendation. In the information domain, we have the interaction data between users and items. Let u and $\mathcal{U}_1 = \{u_t\}_{t=1}^{M_1}$ denote a user and the whole user set of the information domain, respectively; similarly, we use i and $\mathcal{I} = \{i_t\}_{t=1}^{N}$ to denote an item and the whole item set, respectively. The edges between users and items denote their interactions, $\mathcal{Y} = \{y_{ui}\}$, which can be real-valued explicit ratings or binary 0/1 implicit feedback. Traditional collaborative filtering algorithms can then be performed on the user-item interaction data.

In addition to the ID that distinguishes a user or an item, most information-domain sites also associate them with abundant side information, which can help to capture users' preferences and item properties better. For example, in Trip.com, the user may choose the travel tastes of {*luxury travel, art lover*} in her profile; while the item *Marina Bay Sands* is tagged most with travel modes {*luxury travel, family travel, nightlife*}. We term these associated information as *attributes*, most of which are discrete categorical variables for the web domain [10]. Formally, we denote g and $\mathcal{G} = \{g_t\}_{t=1}^{V}$ as an attribute and the whole attribute set, respectively; for a user u and an item i, we can then construct the associated attribute set as $\mathcal{G}_u = \{g_1^u, \cdots, g_{V_u}^u\} \subset \mathcal{G}$ and $\mathcal{G}_i = \{g_1^i, \cdots, g_{V_i}^i\} \subset \mathcal{G}$, respectively.

In the social domain, we have social connections between users, such as the undirected friendship or directed follower/followee relations. We denote a social user as u', all users of the social domain as $\mathcal{U}_2 = \{u_t'\}_{t=1}^{M_2}$, and all social connections as $\mathcal{S} = \{s_{u'u''}\}$. We define the bridge users as the overlapping users between the information domain and social domain. These bridge users can be expressed as $\mathcal{U} = \mathcal{U}_1 \cap \mathcal{U}_2$. In a social network, a user's behaviours and preferences can be propagated along the social connections to influence her friends. As such, these bridge users play a pivotal role in addressing the cross-domain social recommendation problem, which is formally defined as:

Input: An information domain with $\{\mathcal{U}_1, \mathcal{I}, \mathcal{Y}, \mathcal{G}_u, \mathcal{G}_i\}$; a social domain with $\{\mathcal{U}_2, \mathcal{S}\}$; and $\mathcal{U}_1 \cap \mathcal{U}_2$ is nonempty.

Output: A personalized ranking function for each user u' of the social domain $f_{u'} : \mathcal{I} \to \mathbb{R}$, which maps each item of the information domain to a real number.

It is noted that there indeed exist sparse and weak user-item interactions in SNSs as aforementioned. However, we simplify this scenario of cross-domain social recommendation by only emphasizing the social connections in SNSs and leaving the exploration of weak interactions as the future work.
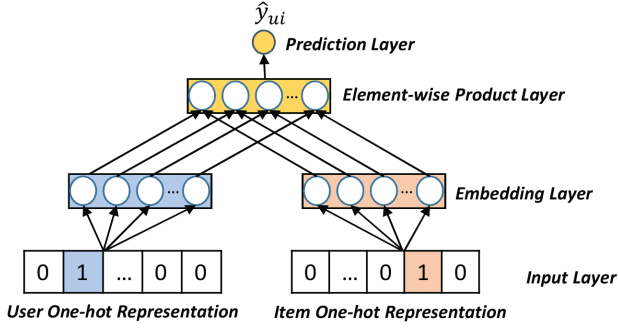
Figure 2: MF as a shallow neural network model.

2.2 Factorization Model

Collaborative filtering (CF) is the key technique for personalized recommendation systems. It exploits user-item interactions by assuming that similar users would have similar preference on items. Model-based CF approaches [1, 33] achieve this goal by describing the interaction data with an underlying model, for which the holistic goal is to build:

$$\widehat{y}_{ui} = f_\Theta(u, i), \tag{1}$$

where f denotes the underlying model with parameters Θ, and $\widehat{y}_{ui}$ denotes the predicted score for a user-item interaction y_{ui}. Matrix factorization (MF) is one of the simplest yet effective models for the recommendation task, which characterizes a user or an item with a latent vector, modelling a user-item interaction as the inner product of their latent vectors:

$$f_{MF}(u, i|\mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u^\top \mathbf{q}_i = \sum_{k=1}^{K} p_{uk} q_{ik}, \tag{2}$$

where $\mathbf{p}_u \in \mathbb{R}^K$ and $\mathbf{q}_i \in \mathbb{R}^K$ are model parameters denoting the latent vector (*aka.* representation) for user u and item i, respectively.

Despite its effectiveness, we note that MF's expressiveness can be limited by the use of the inner product operation to model a user-item interaction. To illustrate this, we present a neural network view of the MF model. As shown in Figure 2, we feed the one-hot representation of user/item ID into the architecture, and project them with a fully connected embedding layer. By feeding the user/item embedding vectors into the element-wise product layer, we obtain a hidden vector $\mathbf{h} = \{p_{uk} q_{ik}\}$. If we directly project $\mathbf{h}$ into the output score, we can exactly recover the MF model. As such, MF can be deemed as a shallow neural network with one hidden layer only. Based on this connection, we argue that there are two key limitations of MF-based approaches for cross-domain social recommendation:

- First, MF only considers the simple two-way interaction between a user and an item, by assuming that their cross latent factors (*i.e.,* $\mathbf{p}_u$ and $\mathbf{q}_i$) are independent of each other. However, such an independence assumption can be insufficient to model real-world data, which usually have complex and non-linear underlying structures [10, 15].
- The case can be even worse if we take the attributes into account. A typical way to extend MF with side attributes is SVDfeature, *i.e.,* by summing attribute embedding vectors with user/item

embedding vector. As a result, the rich correlations among users, items, and attributes are unintentionally ignored.

Our proposed NSCR solution addresses the above limitations of MF by 1) using a deep learning scheme to capture the higher-order correlations between user and item latent factors, and 2) devising a pairwise pooling operation to efficiently model the pair-wise correlations among users, items, and attributes.

3 OUR NSCR SOLUTION

The goal of cross-domain social recommendation is to select relevant items from the information domain for social users. Under the paradigm of embedding-based methods (*aka.* representation learning), the key for addressing the task is on how to project items (of the information domain) and users (of the social domain) into the same embedding space. A generic solution is the factorization machine (FM) [20, 21], which merges the data from the two domains by an early fusion; that is, constructing the predictive model by incorporating social users as the input features. While the solution sounds reasonable conceptually, the problem is that the training instances which can incorporate social users are only applicable to the bridge users, which can be very few for real-world applications. As such, the generic recommender solution FM can suffer severely from the problem of insufficient bridge users.

To address the challenge of insufficient bridge users, we propose a new framework that separates the embedding learning process of each domain. By enforcing the two learning processes to share the same embeddings for bridge users, we can ensure that items and social users are in the same embedding space. Formally, we devise the optimization framework as:

$$\mathcal{L} = \mathcal{L}_I(\Theta_I) + \mathcal{L}_S(\Theta_S), \tag{3}$$

where $\mathcal{L}_I$ (or $\mathcal{L}_S$) denotes the objective function of the information domain (or social domain) learning with parameters Θ_I (or Θ_S), and most importantly, $\Theta_I \cap \Theta_S$ are nonempty denoting the shared embeddings of bridge users.

By separating the learning process for two domains, we allow the design of each component to be more flexible. Specially, we can apply any collaborative filtering solution for $\mathcal{L}_I$ to learn from user-item interactions, and utilize any semi-supervised learning technique for $\mathcal{L}_S$ to propagate the embeddings of bridge users to non-bridge users. In the remainder of this section, we first present our novel neural collaborative ranking solution for $\mathcal{L}_I$, followed by the design of social learning component $\mathcal{L}_S$. Lastly, we discuss how to optimize the joint objective function.

3.1 Learning of Information Domain

To estimate the parameters for a CF model from user-item interaction data, two types of objective functions — point-wise [1, 11] and pair-wise [2, 21, 26] — are most commonly used. The point-wise objective functions aim to minimize the loss between the predicted score and its target value. Here, to tailor our solution for both implicit feedback and the personalized ranking task, we adopt the pair-wise ranking objective functions.

Formally, we denote an observed user-item interaction as $y_{ui} = 1$, otherwise $y_{ui} = 0$. Instead of forcing the prediction score $\hat{y}_{ui}$ to be close to y_{ui}, ranking-ware objective functions concern the relative

order between the pairs of observed and unobserved interactions:

$$\mathcal{L}_I = \sum_{(u,i,j) \in O} \mathcal{L}(y_{uij}, \hat{y}_{uij}), \quad (4)$$

where $y_{uij} = y_{ui} - y_{uj}$ and $\hat{y}_{uij} = \hat{y}_{ui} - \hat{y}_{uj}$; O denotes the set of training triplets, each of which comprises of a user u, an item i of observed interactions (i.e., $y_{ui} = 1$), and an item j of unobserved interactions (i.e., $y_{ui} = 0$). An ideal model should rank all (i, j) item pairs correctly for every user. To implement the ranking hypotheses, we adopt the regression-based loss [26]:

$$\mathcal{L}_I = \sum_{(u,i,j) \in O} (y_{uij} - \hat{y}_{uij})^2 = \sum_{(u,i,j) \in O} (\hat{y}_{ui} - \hat{y}_{uj} - 1)^2. \quad (5)$$

Note that other pair-wise ranking functions can also be applied, such as the bayesian personalized ranking (BPR) [2, 21] and contrastive max-margin loss [23]. In this work, we use the regression-based ranking loss as a demonstration for our NSCR, and leave the exploration of other choices as the future work.

3.1.1 **Attribute-aware Deep CF Model**. Having established the optimization function for learning from information domain, we now present our attribute-aware deep collaborative filtering model to estimate a user-item interaction $\hat{y}_{ui}$. Figure 3 illustrates its architecture, which is a multi-layered feed-forward neural network. We elaborate its design layer by layer.

Input Layer. The input to the model is a user u, an item i, and their associated attributes G_u and G_i. We transform them into barbarized sparse vectors with one-hot encoding, where only the non-zero binary features are recorded.

Embedding Layer. The embedding layer maps each non-zero feature into a dense vector representation. As we have four types of features here, we differentiate them with different symbols: $\mathbf{u}$, $\mathbf{i}$, $\mathbf{g}_t^u$, and $\mathbf{g}_t^i$ denote the K-dimensional embedding vector for user u, item i, user attribute g_t^u, and item attribute g_t^i, respectively.

Pooling Layer. The output of the embedding layer is a set of embedding vectors to describe user u and item i, respectively. As different users (items) may have different number of attributes, the size of the embedding vector set may vary for different inputs. To train a neural network of fixed structure, it is essential to convert the set of variable-length vectors to a fixed-length vector, i.e., the pooling operation.

The most commonly used pooling operations in neural network modelling are average pooling and max pooling. However, we argue that such simple operations are insufficient to capture the interaction between users/items and attributes. For example, the average pooling assumes a user and her attributes are linearly independent, which fails to encode any correlation between them in the embedding space. To tackle the problem, we consider to model the pairwise correlation between a user and her attributes, and all nested correlations among her attributes:

$$\mathbf{p}_u = \varphi_{pairwise}(\mathbf{u}, \{\mathbf{g}_t^u\}) = \sum_{t=1}^{V_u} \mathbf{u} \odot \mathbf{g}_t^u + \sum_{t=1}^{V_u} \sum_{t'=t+1}^{V_u} \mathbf{g}_t^u \odot \mathbf{g}_{t'}^u, \quad (6)$$

where $\odot$ denotes the element-wise product of two vectors. We term it as *pairwise pooling*, which is originally inspired from the design of factorization machines [10, 19]. By applying pairwise pooling
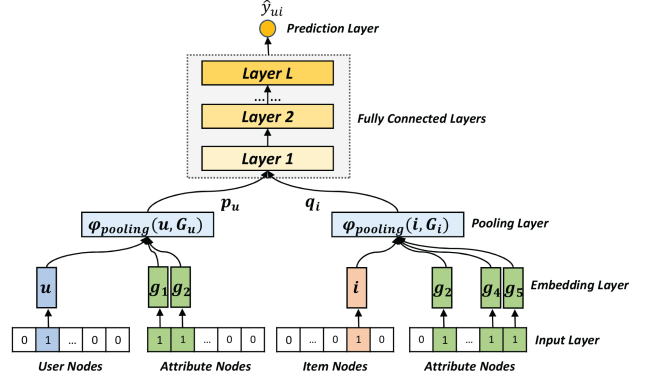

Figure 3: Illustration of our Attributed-aware Deep CF model for estimating an user-item interaction.

on the item counterpart, we can similarly model the pair-wise correlation between an item and its attributes:

$$\mathbf{q}_i = \varphi_{pairwise}(\mathbf{i}, \{\mathbf{g}_t^i\}) = \sum_{t=1}^{V_i} \mathbf{i} \odot \mathbf{g}_t^i + \sum_{t=1}^{V_i} \sum_{t'=t+1}^{V_i} \mathbf{g}_t^i \odot \mathbf{g}_{t'}^i. \quad (7)$$

It is worth pointing out that although pairwise pooling models the correlation between each pair of features, it can be efficiently computed in linear time — the same time complexity with average/max pooling. To show the linear time complexity of evaluating pairwise pooling, we reformulate Eqn.(6) as,

$$\mathbf{p}_u = \frac{1}{2}\left[\left(\mathbf{u} + \sum_{t=1}^{V_u} \mathbf{g}_t^u\right) \odot \left(\mathbf{u} + \sum_{t=1}^{V_u} \mathbf{g}_t^u\right) - \mathbf{u} \odot \mathbf{u} - \sum_{t=1}^{V_u} \mathbf{g}_t^u \odot \mathbf{g}_t^u\right], \quad (8)$$

which can be computed in $O(KV_u)$ time. This is a very appealing property, meaning that the benefit of pairwise pooling in modelling all pair-wise correlations does not involve any additional cost, as compared to the average pooling that does not model any correlation between input features.

Hidden Layers: Above the pairwise pooling is a stack of full connected layers, which enable us to capture the nonlinear and higher-order correlations among users, items, and attributes. Inspired by the neural network view of matrix factorization (cf. Figure 2), we first merge user representation $\mathbf{p}_u$ and item representation $\mathbf{q}_i$ with an element-wise product, which models the two-way interaction between u and i. We then place a multi-layer perceptron (MLP) above the element-wise product. Formally, the hidden layers are defined as:

$$\begin{cases} \mathbf{e}_1 = \sigma_1(\mathbf{W}_1(\mathbf{p}_u \odot \mathbf{q}_i) + \mathbf{b}_1) \\ \mathbf{e}_2 = \sigma_2(\mathbf{W}_2\mathbf{e}_1 + \mathbf{b}_2) \\ \cdots\cdots \\ \mathbf{e}_L = \sigma_L(\mathbf{W}_L\mathbf{e}_{L-1} + \mathbf{b}_L) \end{cases}, \quad (9)$$

where $\mathbf{W}_l$, $\mathbf{b}_l$, σ_l, and $\mathbf{e}_l$ denote the weight matrix, bias vector, activation function, and output vector of the l-th hidden layers, respectively. As for the activation function in each hidden layer, we opt for Rectifier (ReLU) unit, which is more biologically plausible and proven to be non-saturated. Regarding the structure of hidden layers, common choices include the tower [4, 11], constant, and diamond, among others. In this work, we simply set all hidden

layers have the same size, leaving the further tuning of the deep structure as the future work.

Prediction Layer: At last, the output vector of the last hidden layer $\mathbf{e}_L$ is transformed to the prediction score:

$$\hat{y}_{ui} = \mathbf{w}^\top \mathbf{e}_L, \qquad (10)$$

where $\mathbf{w}$ represents the weight vector of the prediction layer.

Note that we have recently proposed a neural factorization machine (NFM) model [10], which similarly uses a pairwise pooling operation to model the interaction among features. We point out that the main architecture difference is in our separated treatment of the user and item channel, where each channel can essentially be seen as an application of NFM on the user/item ID and attributes.

3.2 Learning of Social Domain

With the above neural collaborative ranking solution, we obtain an attribute-aware representation $\mathbf{p}_u$ and $\mathbf{q}_i$ for each user and item, respectively. To predict the affinity score of a social user to an item of the information domain, we need to also learn an representation for the social user in the same latent space of the information domain. We achieve this goal by propagating $\mathbf{p}_u$ from bridge users to representations for non-bridge users of the social domain. The intuition for such representation propagation is that, if two users are strongly connected (*e.g.,* close friends with frequent interactions), it is likely that they have the similar preference on items; as such, they should have similar representations in the latent space. This suits well the paradigm of graph regularization [7, 9, 28, 29] (*aka.* semi-supervised learning on graph), which has two components:

Smoothness: The smoothness constraint implies the structural consistency — the nearby vertices of a graph should not vary much in their representations. Enforcing smoothness constraint in our context of social domain learning will propagate a user's representation to her neighbors, such that when a steady state reaches, all vertices should have been placed in the same latent space. The objective function for smoothness constraint is defined as:

$$\theta(\mathcal{U}_2) = \frac{1}{2} \sum_{u',u''\in\mathcal{U}_2} s_{u'u''} \left\| \frac{\mathbf{p}_{u'}}{\sqrt{d_{u'}}} - \frac{\mathbf{p}_{u''}}{\sqrt{d_{u''}}} \right\|^2, \qquad (11)$$

where $s_{u'u''}$ denotes the strength of social connection between u' and u'', and $d_{u'}$ (or $d_{u''}$) denotes the outdegree of u' (or u'') for normalization purpose. It is worth noting that the use of normalization is the key difference with the social regularization used by [16, 36], which does not apply any normalization on the smoothness constraint. As pointed out by He *et al.* [9], the use of normalization helps to suppress the impact of popular vertices, which can lead to more effective propagation. We empirically verify this point in Section 4.3.

Fitting: The fitting constraint implies the latent space consistency across two domains — the bridge users' representations should be invariant and act as the anchors across domains. Towards this end, we encourage the two representations of the same bridge users to be close to each other. The objective function for fitting constraint is defined as,

$$\theta(\mathcal{U}) = \frac{1}{2} \sum_{u'\in\mathcal{U}} \left\| \mathbf{p}_{u'} - \mathbf{p}_{u'}^{(0)} \right\|^2, \qquad (12)$$

where for each bridge user u', $\mathbf{p}_{u'}$ (or $\mathbf{p}_{u'}^{(0)}$) is her representation of the SNS (or information domain). As such, the fitting constraint essentially acts as the bridges connecting the two latent spaces.

Lastly, we combine the smoothness constraint with the fitting constraint and obtain the objective function of the social domain learning as,

$$\mathcal{L}_S = \theta(\mathcal{U}_2) + \mu\theta(\mathcal{U}), \qquad (13)$$

where μ is a positive parameter to control the tradeoff between two constraints.

3.2.1 **Prediction for Social Users.** With the representations of social users and items (*i.e.,* $\mathbf{p}_{u'}$ and $\mathbf{q}_i$) at hand, we can feed them into the fully connected layers as Eqn.(9) shows and utilize the prediction layer as Eqn.(10) displays. At last, we can obtain the predicted preference $\widehat{y}_{u'i}$, as follows,

$$\begin{cases} \mathbf{e}_1 = \sigma_1(\mathbf{W}_1(\mathbf{p}_{u'} \odot \mathbf{q}_i) + \mathbf{b}_1) \\ \cdots\cdots \\ \mathbf{e}_L = \sigma_L(\mathbf{W}_L\mathbf{e}_{L-1} + \mathbf{b}_L) \\ \widehat{y}_{u'i} = \mathbf{w}^\top\mathbf{e}_L \end{cases}. \qquad (14)$$

3.3 Training

We adopt the alternative optimization strategy on Eqn.(3) since it can emphasize exclusive characteristics within individual domains. In the information domain, we employ stochastic gradient descent SGD) to train the attribute-aware NSCR in the mini-batch mode and update the corresponding model parameters. In particular, we first sample a batch of observed user-item interactions (u, i) and adopt negative sampling [11] to randomly select an unobserved item j for each (u, i). We then generate a triplet (u, i, j). Following that, we take a gradient step to optimize the loss function $\mathcal{L}_I$ in Eqn.(5). As such, we obtain the enhanced representations of users. In the SNS, we feed the enhanced representations of bridge users into our graph Laplacian to update all representations of social users. Towards this end , we can simplify the derivative of $\mathcal{L}_S$ regarding user representation $\mathbf{P}$ and then obtain the close-form solution as,

$$\mathbf{P} = \frac{\mu}{1+\mu} \left(\mathbf{I} - \frac{1}{1+\mu} \mathbf{D}^{-\frac{1}{2}} \mathbf{S} \mathbf{D}^{-\frac{1}{2}} \right)^{-1} \mathbf{P}^{(0)}, \qquad (15)$$

where $\mathbf{P}^{(0)}$ is the embedding of social users, which includes the updated representations of bridge users from NSCR part; $\mathbf{S}$ and $\mathbf{D}$ are the similarity matrix and diagonal degree matrix of social users, respectively, whereinto $S_{u'u''} = s_{u'u''}$ and $D_{u'u'} = d_{u'}$. Thereafter, we view the newly updated representations of bridge users as the next initialization for the bridge users in NSCR. We repeat the above procedures to approximate the model parameter set Θ. As for the regularization term in Eqn.(3), we omit it since we utilize *dropout* technique in neural network modeling to avoid overfitting.

Dropout: Dropout is an effective solution to prevent deep neural networks from overfitting. The idea is to randomly drop part of neurons during training. As such, only part of the model parameters, which contribute to the final ranking, will be updated. In our neural CR model, we propose to adopt dropout on the pairwise pooling layer. In particular, we randomly drop ρ of $\mathbf{p}_u$ and $\mathbf{q}_i$, whereinto ρ is the dropout ratio. Analogous to the pooling layer, we also conduct dropout on each hidden layer.

4 EXPERIMENTS

To comprehensively evaluate our proposed method, we conducted experiments to answer the following research questions:

- **RQ1:** Can our NSCR approach outperform the state-of-the-art recommendation methods for the new cross-domain social recommendation task?
- **RQ2:** How do different hyper-parameter settings (*e.g.,* the dropout ratio and tradeoff parameters) affect NSCR?
- **RQ3:** Are deeper hidden layers helpful for learning from user-item interaction data and improving the performance of NSCR?

4.1 Data Description

To the best of our knowledge, there is no available public benchmark dataset that fits the task of cross-domain social recommendation. As such, we constructed the datasets by ourselves. We treated Trip.com as the information domain, Facebook and Twitter as the social domains. In Trip.com, we initially compiled $6,532$ active users, who had at least 5 ratings over $2,952$ items (*e.g., gardens by the bay* in Singapore and *eiffel tower* in Pairs). We transformed their $93,998$ ratings into binary implicit feedback as ground truth, indicating whether the user has rated the item. Moreover, we collected 19 general categories regarding the travel mode (*e.g., adventure travel, business travel,* and *nightlife*) and used them as the attributes of users and items. Subsequently, we parsed the users' profiles to identify their aligned accounts in Facebook and Twitter, inspired by the methods in [17, 24]. We obtained 858 and 502 bridge users for Facebook and Twitter, respectively. Thereafter, we crawled the public friends or followers of each bridge user to reconstruct the social networks, resulting in $177,042$ Facebook users and $106,049$ Twitter users. However, the original social data are highly sparse, where most non-bridge users have only one friend, making it ineffective to propagate users' preferences. To ensure the quality of the social data, we performed a modest filtering on the data, retraining users with at least two friends. This results in a subset of the social data that contains $7,233$ Twitter users with $42,494$ social connections and $8,196$ Facebook users with $49,156$ social connections. The statistics of the datasets are summarized in Table 1.

4.2 Experimental Settings

Evaluation Protocols: Given a social user, each method generates an item ranking list for the user. To assess the ranking list, we adopted two popular IR metrics, *AUC* and *recall*, to measure the quality of preference ranking and top-N recommendation.

- **AUC:** Area under the curve (AUC) [12, 21] measures the probability that a recommender system ranks a positive user-item interaction higher than negative ones:

$$AUC = \frac{\sum_{i \in \mathcal{I}_u^+} \sum_{j \in \mathcal{I}_u^-} \delta(\widehat{y}_{uij} > 0)}{|\mathcal{I}_u^+||\mathcal{I}_u^-|}, \tag{16}$$

where $\mathcal{I}_u^+ = \{i|y_{ui} = 1\}$ and $\mathcal{I}_u^- = \{j|y_{uj} = 0\}$ denote the sets of relevant (observed) item i and irrelevant (unobserved) item j for user u, respectively; and δ is the count function returning 1 if $\widehat{y}_{uij} > 0$ and 0 otherwise. Below we report the averaged AUC for all testing users.

Table 1: Statistics of the complied datasets. The social user set includes the bridge users.

Information Domain	User#	Item#	Interaction#
Trip.com	$6,532$	$2,952$	$93,998$
SNSs	**Bridge User#**	**Social User#**	**Social Connection#**
Twitter	502	$7,233$	$42,494$
Facebook	858	$8,196$	$49,156$

- **R@K:** Recall@K considers the relevant items within the top K positions of the ranking list. A higher recall with lower K indicates a better recommender system, which can be defined as,

$$R@K = \frac{|\mathcal{I}_u^+ \cap \mathcal{R}_u|}{|\mathcal{I}_u^+|}, \tag{17}$$

where $\mathcal{R}_u$ denotes the set of the top-K ranked items for the given user u. Analogous to AUC, we report the average $R@5$ for all testing users.

By learning representations for social users and information-domain items together, our NSCR is capable of recommending items for both bridge and non-bridge users. However, due to the limitation of our static datasets, it is difficult for us to evaluate the recommendation quality for non-bridge users, since they have no interaction on the information-domain items. As such, we rely on the bridge users for evaluating the performance. Following the common practice in evaluating a recommender algorithm [11, 21], we holdout the latest 20% interactions of a bridge user as the test set. To tune hyper-parameters, we further randomly holdout 20% interactions from a bridge user's training data as the validation set. We feed the remaining bridge users, all the non-bridge users in SNSs, and the remaining user-item interactions in the information domains into our framework for training.

Baselines: To justify the effectiveness of our proposal, we study the performance of the following methods:

- **ItemPop:** This method ranks items base on their popularity, as judged by the number of interactions. It is a non-personalized method that benchmarks the performance of a personalized system [21].
- **MF:** This is the standard matrix factorization model that leverages only user–item interactions of the information domain for recommendation (*cf.* Eqn.(2)).
- **SFM:** Factorization machine [19] is a generic factorization model that is designed for recommendation with side information. We construct the input feature vector by using one-hot encoding on the ID and attributes of users and items. To adjust FM for modelling social relations, we further plug a (bridge) user's friends into the input feature vector, dubbed this enhanced model as Social-aware FM (SFM).
- **SR:** This [16] is a state-of-the-art factorization method for social recommendation. It leverages social relations to regularize the latent vectors of friends to be similar. To incorporate attributes into their method, we adjust the similarity of two users based on their attribute sets, which leads to better performance.

Note that for all model-based methods, we optimize them with the same pair-wise ranking function of Eqn.(5) for a fair comparison on the model's expressiveness. To explore the efficacy of attributes, we further explore variants that remove attribute modelling from SFM, SR, and NSCR, named as SFM-a, SR-a, and NSCR-a, respectively.

Table 2: Performance comparison between all the methods, when the embedding size= 64 and significance test is based on AUC.

Datasets	Twitter-Trip			Facebook-Trip		
Methods	AUC	R@5	p-value	AUC	R@5	p-value
ItemPop	0.7193	0.0164	3e-5	0.7439	0.0249	8e-6
MF	0.8285	0.0375	3e-4	0.8596	0.0821	1e-4
SFM	0.8832	0.0492	2e-3	0.8908	0.0856	1e-3
SR	0.9013	0.0747	9e-3	0.9267	0.1433	4e-2
NSCR	**0.9222**	**0.0807**	-	**0.9390**	**0.1466**	-

Parameter Settings: We implemented our proposed framework on the basis of Tensorflow[2], which will be made publicly available, as well as our datasets. For all the neural methods, we randomly initialized model parameters with a Gaussian distribution, whereinto the mean and standard deviation is 0 and 0.1, respectively. The mini-batch size and learning rate for all methods was searched in [128, 256, 512, 1024] and [0.0001, 0.0005, 0.001, 0.05, 0.1], respectively. We selected Adagrad as the optimizer. Moreover, we empirically set the size of hidden layer same as the embedding size (the dimension of the latent factor) and the activation function as ReLU. Without special mention, we employed two hidden layers for all the neural methods, including SFM, SR, and NSCR. We randomly generated ten different initializations and feed them into our NSCR. For other competitors, the initialization procedure is analogous to ensure the fair comparison. Thereafter, we performed paired t-test between our model and each of baselines over 10-round results.

4.3 Performance Comparison (RQ1)

We first compare the recommendation performance of all the methods. We then purpose to justify how the social modelling and the attribute modelling affect the recommendation performance.

Overall Comparison: Table 2 displays the performance comparison *w.r.t.* AUC and R@5 among the recommendation methods on Twitter-Trip and Facebook-Trip datasets, where the embedding size is 64 for all the methods. We have the following findings:

- ItemPop achieves the worst performance, indicating the necessity of modelling users' personalized preferences, rather than just recommending popular items to users. As for MF, its unsatisfied performance reflects that the independence assumption is insufficient to capture the complex and non-linear structure of user-item interactions.
- NSCR substantially outperforms the state-of-the-art methods, SFM and SR. We further conduct one-sample t-tests, verifying that all improvements are statistically significant with *p*-value < 0.05. It justifies the effectiveness of our proposed framework.
- The performance on Twitter-Trip clearly underperforms that of Facebook-Trip. It is reasonable since more bridge users are available in Facebook, which can lead to better embedding learning in SNSs. It again verifies the significance of the bridge users.

Effect of Social Modelling: To analyze the effect of social modelling, we only consider the variants, SFM-a, SR-a, and NSCR-a.

[2]https://www.tensorflow.org.

Figure 4 presents the performance comparison *w.r.t.* the number of latent factors on two datasets. We have the following observations.

- ItemPop and MF perform worst since neither of them considers the social connections from SNSs. It highlights the necessity of social modelling in cross-domain social recommendation.
- Clearly, NSCR-a significantly outperforms SFM-a and SR-a by a large margin. Formally, in terms of AUC, the relative improvement over SFM-a and SR-a, on average, is 3.19% and 1.01% respectively. While SFM-a considers modelling the social connections, it treats these connections as ordinary features, overlooking the exclusive characteristics of social networks. This leads to the poor expressiveness of the social users' embedding. On the contrary, SR-a and NSCR-a emphasizes the social modelling via the effective social regularization.
- Lastly, NSCR-a shows consistent improvements over SR-a, admitting the importance of the normalized graph Laplacian. It again verifies that the normalized graph Laplacian can suppress the popularity of friends and further prevent the social modelling from being dominated by popular social users.

Effect of Attribute Modelling: As Figure 5 demonstrates, we verify the substantial influence of attribute modelling and the effectiveness of our pairwise pooling operation. Due to the poor performance of ItemPop and MF, they are omitted. Jointly analyzing the performance of all the methods and their variants, we find that,

- For all methods, modelling user/item attributes can achieve significant improvements. By leveraging the similarity of users' attributes, SR enriches the pairwise similarity of any two users and strengthens their connections; meanwhile, SFM can model the correlations of user-attribute, item-attribute, and attribute-attribute, and accordingly enhances the user-item interactions. Benefiting from the pairwise pooling operation, NSCR can encode the second-order interactions between user/item and attributes and boost the representation learning. The significance of attribute is consistent with [34].
- Varying the embedding size, we can see that large embedding may cause overfitting and degrade the performance. In particular, the optimal embedding size is 64 and 32 for AUC and R@5, respectively. It indicates that the setting of embedding size can effect the expressiveness of our model.

4.4 Study of NSCR (RQ2)

In this subsection, we empirically study the convergence of NSCR and then purpose to analyse the influences of several factors, such as dropout ratio and tradeoff parameter, on our framework.

Convergence: We separately present the training loss and the performance w.r.t. AUC and R@5 of each iteration in Figures 6(a), 6(b), and 6(c). Jointly observing these Figures, we can see that training loss of NSCR gradually decreases with more iterations, whereas the performance is generally improved. This indicates the rationality of our learning framework. Moreover, the most effective updates occurs in the first 20 iterations, which indicates that effectiveness of our learning framework. As Figure 6(c) shows, the performance regarding R@5 fluctuates markedly over the iteration times, while that regarding AUC is quite stable. It is reasonable since R@5 only considers the top-5 results rather than the relative order as AUC defined.
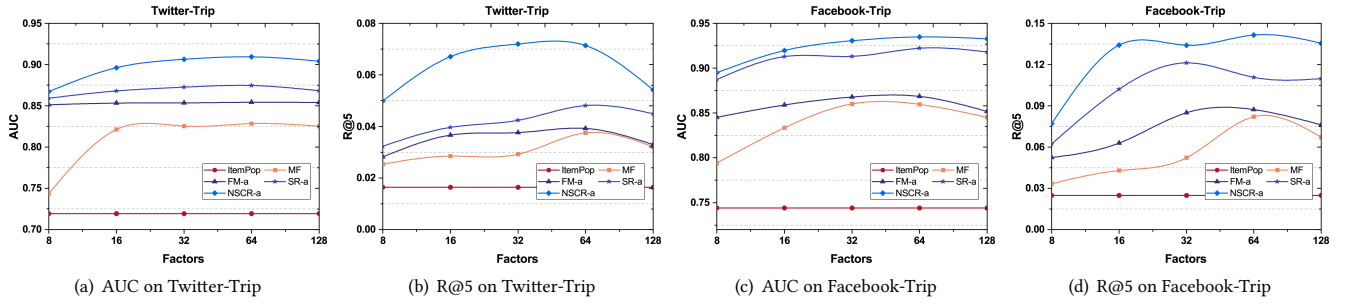
(a) AUC on Twitter-Trip   (b) R@5 on Twitter-Trip   (c) AUC on Facebook-Trip   (d) R@5 on Facebook-Trip

Figure 4: Performance comparison of AUC and R@5 *w.r.t.* the embedding size on Twitter-Trip and Facebook-Trip datasets.

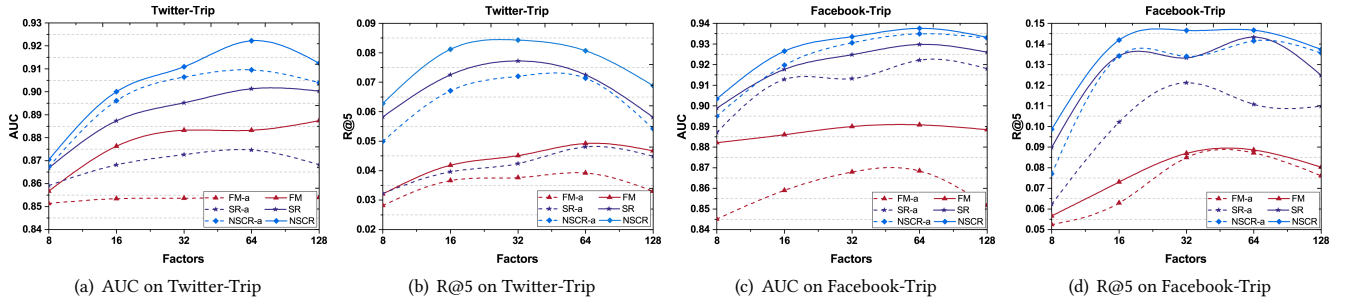(a) AUC on Twitter-Trip   (b) R@5 on Twitter-Trip   (c) AUC on Facebook-Trip   (d) R@5 on Facebook-Trip

Figure 5: Performance comparison of AUC and R@5 *w.r.t.* the embedding size on Twitter-Trip and Facebook-Trip datasets.

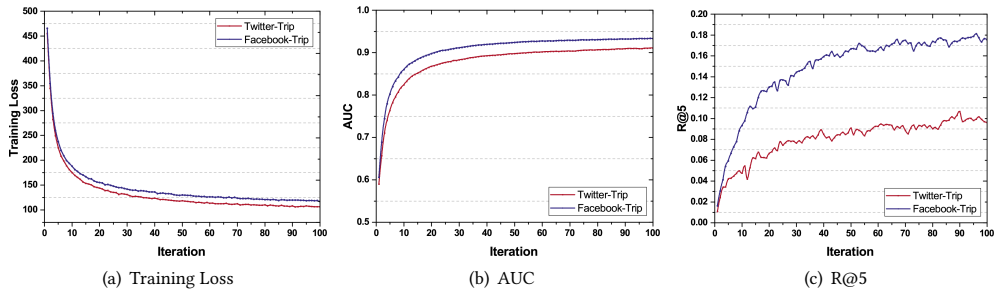(a) Training Loss   (b) AUC   (c) R@5

Figure 6: Training loss and recommendation performance regarding AUC and R@5 *w.r.t.* the number of iterations.

(a) AUC vs. dropout ratio ρ (b) R@5 vs. dropout ratio ρ (c) AUC vs. tradeoff parameter μ (d) R@5 vs. tradeoff parameter μ
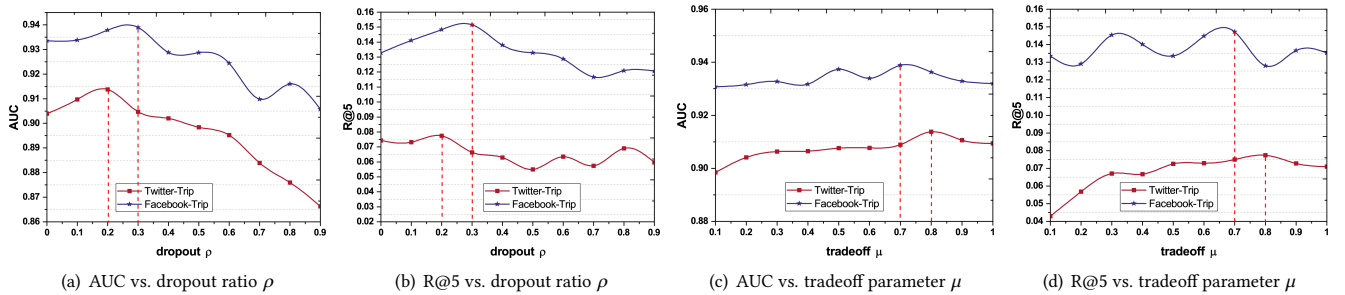
Figure 7: Performance comparison of AUC and R@5 *w.r.t.* the dropout ratio ρ and tradeoff parameter μ on Twitter-Trip and Facebook-Trip datasets.

Impact of Dropout: We employ the dropout technique in NSCR to prevent our model from overfitting, instead of regularizing model parameters. Figures 7(a) and 7(b) present the performance *w.r.t.* AUC and R@5 of NSCR-0 by varying the dropout ratio ρ on the pairwise pooling layer, respectively. As we can see, when dropout ratio equals to 0, NSCR-0 suffers severely from overfitting. Moreover, using a dropout ratio of 0.3 and 0.2 leads to

the best performance on Twitter-Trip and Facebook-Trip datasets, respectively. However, when the optimal dropout ratio exceeds the optimal settings, the performance of NSCR-0 greatly decreases, which suffers from insufficient information. This highlights the significance of using dropout, which can be seen as ensembling multiple sub-models [25].

Table 3: Recommendation performance of NSCR with different hidden layers.

Metrics	AUC			R@5		
Factors	NSCR-0	NSCR-1	NSCR-2	NSCR-0	NSCR-1	NSCR-2
Twitter-Trip						
8	0.8598	0.8630	**0.8704**	0.0585	0.0604	**0.0628**
16	0.8883	0.8984	**0.9026**	0.0738	0.0672	**0.0812**
32	0.9018	0.9056	**0.9109**	0.0723	0.0742	**0.0843**
64	0.9138	0.9175	**0.9222**	0.0717	0.0697	**0.0725**
128	0.9003	0.9034	**0.9125**	0.0519	0.0653	**0.0688**
Facebook-Trip						
8	0.8978	0.8922	**0.9034**	0.0860	0.0872	**0.0986**
16	0.9165	0.9197	**0.9265**	0.1048	0.1388	**0.1419**
32	0.9303	0.9322	**0.9335**	0.1441	**0.1486**	0.1465
64	0.9337	0.9376	**0.9390**	0.1353	0.1359	**0.1466**
128	0.9270	0.9310	**0.9332**	0.1168	0.1304	**0.1373**

Impact of Tradeoff Parameter: There is one positive parameter μ in the social modelling, which can capture the tradeoff between the fitting regularizer and the normalized graph Laplacian, as Eqn.(15) shows. Figures 7(c) and 7(d) present the performance *w.r.t.* . AUC and R@5, respectively. As we can see, setting μ of 0.8 and 0.7 can lead to the optimal performance on Twitter-Trip and Facebook-Trip datasets, respectively. And the performance of NSCR-0 changes within small ranges nearby the optimal settings. It justifies that our model is relatively insensitive to the parameter around its optimal configuration.

4.5 Impact of Hidden Layer (RQ3)

To capture the complex and non-linear inherent structure of user-item interactions, we employ the a deep neural network for our task. It is curious whether NSCR can benefit from the deep architecture. Towards this end, we further investigate NSCR with different number of hidden layers. As it is computationally expensive to tune the dropout ratio ρ for each hidden layer, we simply apply the same settings for all layers. The empirical results on two datasets are summarized in Table 3 whereinto NSCR-2 indicates the NSCR method with two hidden layers (besides the embedding layer and prediction layer), and similar notations for others. We have the following observations:

- In most cases, stacking more hidden layers is helpful for the recommendation performance. NSCR-2 and NSCR-1 achieve consistent improvement over NSCR-0, which has no hidden layers and directly projects the embedding to the prediction layer. We attributed the improvement to the high nonlinearity achieved by stacking more hidden layers. Our finding is consistent with [8] and again verifies the deep neural networks have strong generalization ability. However, it is worth mentioning that such a deep architecture needs more time to optimize our framework and easily leads to the overfitting due to the limited training data in our datasets.
- Increasing the width of hidden layers (*i.e.,* the embedding size) from 8 to 64 can improve the performance significantly, as that of increasing their depth. However, with the embedding size of 128, NSCR degrades the performance. It again verifies that using a large number of the embedding size has powerful representation ability [8], but may adversely hurt the generalization of the model (*e.g.,* overfitting the data) [8, 11].

5 RELATED WORK

5.1 Social Recommendation

Social recommendation aims to leverage users' social connections to enhance a recommender system [18, 32]. It works by modelling social influence, which refers to the fact that a user's decision can be affected by her friends' opinions and behaviours. Ma *et al.* [16] propose a social regularization term to enforce social constraints on traditional recommender systems. Based on a generative influence model, the work [31] exploits social influence from friends for item recommendation by leveraging information embedded in the user social network. The authors in [35] utilize social links as complementary data source to mine topic domains and employed domain-specific collaborative filtering to formulate users' interests. More recently, [13] represents a star-structured hybrid graph centered at a user domain, which connects with other item domains, and transfers knowledge on social networks.

It is worth noting that the aforementioned studies are all based on social network relations of an information domain. While in this work, we focus on how to distill useful signal from an external social network (*e.g.,* Facebook and Twitter), so as to improve the recommendation service of any information domain.

5.2 Cross-Domain Recommendation

Distinct from the traditional recommendation methods that focus on data within a single domain, cross-domain recommendation concerns data from multiple domains. A common setting is leveraging the user-item interaction of a related auxiliary domain to improve the recommendation of the target domain. However, existing cross-domain recommendation work has an underlying assumption that the target and auxiliary domains are homogeneous. Depending on [5, 6, 13], they can be divided into two directions. One is assuming that different domains share overlapped user or item sets. The work [22] augments ratings of movies and books for the shared users and accordingly conducts CF. Based on the shared users' latent space, the authors in [3] leveraged cluster-level tensor sharing as a social regularization to bridge the domains. One more step, the authors in [12] formulated a generalized triadic user-item-domain relation over the common users and accordingly to capture domain-specific user factors and item factors. More recently, the authors [5] proposed a multi-view deep learning recommendation system by using auxiliary rich features to represent users from different domains. Without aligned user or item, the other direction is on homogeneous data with the same rating scale. Codebook Transfer [14] represents cluster-level rating patterns between two rating matrices in two related domains. [27] introduces a topic model to recommend authors to collaborate from different research fields.

Despite the compelling success achieved by previous work, little attention has been paid to recommendation across heterogeneous domains. In our settings, the source domain is a social network with user-user relations only, while the target domain is an information domain with user-item interactions. Hence, the auxiliary information is the social friendship, rather than the conventional interaction data. As a result, existing approaches can be hardly applied to this new research problem.

6 CONCLUSION

In this work, we systematically investigated cross-domain social recommendation, a practical task that has rarely been studied previously. Towards this end, we proposed a generic neural social collaborative ranking (NSCR) solution, which seamlessly integrates user-item interactions of the information domain and user-user social relations of the social domain. To validate our solution, we constructed two real-world benchmarks of the travel domain, performing extensive experiments to demonstrate the effectiveness and rationality of our NSCR solution. The key finding of the work is that social signals contain useful cues about users' preference, even if the social signals are from social networks in a different domain. We achieved the goal by leveraging bridge users to unify the relevance signals from the two heterogeneous domains.

Due to our restricted resources in collecting cross-domain data, the result is preliminary. Here we discuss several limitations of the current work, and our plans to address them in future. First, in this work, we studied the recommendation performance of a travel-based information domain only, which is mainly for the ease of accessing the users' account on Facebook/Twitter. This results in a relatively small number of bridge users of our cross-domain datasets. As a future work, we will collect a larger-scale set of data from the more popular information domains, such as E-commence sites, to explore the generalization ability of our solution to other information domains. Second, due to the small number of bridge users, we forwent the study of user cold-start problem, as further holding out bridge users to simulate the cold-start scenario will pose challenge to the stability of evaluation. With a larger-scale cross-domain data, we will study the effectiveness of our solution for cold-start users, as well as the influence of the bridge users' percentage. Moreover, we restricted the SNSs by emphasizing only the social connections and omitting the weak user-item interactions in user-generated-contents. We will consider the weak user-item interaction in both domains to improve the recommendation performance.

REFERENCES

[1] Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. 2017. A Generic Coordinate Descent Framework for Learning from Implicit Feedback. In *WWW*. 1341–1350.

[2] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie and Wei Liu, and Tat-Seng Chua. 2017. Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention. In *SIGIR*.

[3] Wei Chen, Wynne Hsu, and Mong-Li Lee. 2013. Making recommendations from multiple domains. In *SIGKDD*. 892–900.

[4] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *RecSys*. 191–198.

[5] 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*. 278–288.

[6] Aleksandr Farseev, Ivan Samborskii, Andrey Filchenkov, and Tat-Seng Chua. 2017. Cross-Domain Recommendation via Clustering on Multi-Layer Graphs. In *SIGIR*.

[7] Fuli Feng, Liqiang Nie, Xiang Wang, Richang Hong, and Tat-Seng Chua. 2017. Computational social indicators: a case study of Chinese university ranking. In *SIGIR*.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*. 770–778.

[9] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. TriRank: Review-aware Explainable Recommendation by Modeling Aspects. In *CIKM*. 1661–1670.

[10] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. (2017).

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2016. Neural Collaborative Filtering. In *WWW*. 173–182.

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

[13] Meng Jiang, Peng Cui, Xumin Chen, Fei Wang, Wenwu Zhu, and Shiqiang Yang. 2015. Social Recommendation with Cross-Domain Transferable Knowledge. *TKDE* 27, 11 (2015), 3084–3097.

[14] Bin Li, Qiang Yang, and Xiangyang Xue. 2009. Can Movies and Books Collaborate? Cross-Domain Collaborative Filtering for Sparsity Reduction. In *IJCAI*. 2052–2057.

[15] Lizi Liao, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2017. Attributed Social Network Embedding. *arXiv preprint arXiv:1705.04969* (2017).

[16] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *WSDM*. 287–296.

[17] Liqiang Nie, Xuemeng Song, and Tat-Seng Chua. 2016. *Learning from Multiple Social Networks*. Morgan & Claypool Publishers.

[18] Zhaochun Ren, Shangsong Liang, Piji Li, Shuaiqiang Wang, and Maarten de Rijke. 2017. Social Collaborative Viewpoint Regression with Explainable Recommendations. In *WSDM*. 485–494.

[19] Steffen Rendle. 2010. Factorization Machines. In *ICDM*. 995–1000.

[20] Steffen Rendle. 2012. Factorization Machines with libFM. *TIST* 3, 3 (2012), 57:1–57:22.

[21] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. 452–461.

[22] Shaghayegh Sahebi and Peter Brusilovsky. 2013. Cross-Domain Collaborative Recommendation in a Cold-Start Context: The Impact of User Profile Size on the Quality of Recommendation. In *UMAP*. 289–295.

[23] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *NIPS*. 926–934.

[24] Xuemeng Song, Liqiang Nie, Luming Zhang, Mohammad Akbari, and Tat-Seng Chua. 2015. Multiple Social Network Learning and Its Application in Volunteerism Tendency Prediction. In *SIGIR*. 213–222.

[25] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* 15, 1 (2014), 1929–1958.

[26] Gábor Takács and Domonkos Tikk. 2012. Alternating Least Squares for Personalized Ranking. In *RecSys*. 83–90.

[27] Jie Tang, Sen Wu, Jimeng Sun, and Hang Su. 2012. Cross-domain collaboration recommendation. In *SIGKDD*. 1285–1293.

[28] Meng Wang, Weijie Fu, Shijie Hao, Hengchang Liu, and Xindong Wu. 2017. Learning on Big Graph: Label Inference and Regularization with Anchor Hierarchy. *TKDE* 29, 5 (2017), 1101–1114.

[29] Meng Wang, Weijie Fu, Shijie Hao, Dacheng Tao, and Xindong Wu. 2016. Scalable Semi-Supervised Learning by Efficient Anchor Graph Regularization. *TKDE* 28, 7 (2016), 1864–1877.

[30] Xiang Wang, Liqiang Nie, Xuemeng Song, Dongxiang Zhang, and Tat-Seng Chua. 2017. Unifying Virtual and Physical Worlds: Learning Toward Local and Global Consistency. *TOIS* 36, 1 (2017), 4.

[31] Mao Ye, Xingjie Liu, and Wang-Chien Lee. 2012. Exploring social influence for recommendation: a generative model approach. In *SIGIR*. 671–680.

[32] Chao Zhang, Keyang Zhang, Quan Yuan, Luming Zhang, Tim Hanratty, and Jiawei Han. 2016. GMove: Group-Level Mobility Modeling Using Geo-Tagged Social Media. In *SIGKDD*. 1305–1314.

[33] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat-Seng Chua. 2016. Discrete Collaborative Filtering. In *SIGIR*. 325–334.

[34] Hanwang Zhang, Zheng-Jun Zha, Yang Yang, Shuicheng Yan, Yue Gao, and Tat-Seng Chua. 2013. Attribute-augmented semantic hierarchy: towards bridging semantic gap and intention gap in image retrieval. In *MM*. 33–42.

[35] Xi Zhang, Jian Cheng, Ting Yuan, Biao Niu, and Hanqing Lu. 2013. TopRec: domain-specific recommendation through community topic mining in social network. In *WWW*. 1501–1510.

[36] Z. Zhao, H. Lu, D. Cai, X. He, and Y. Zhuang. 2016. User Preference Learning for Online Social Recommendation. *TKDE* 28, 9 (2016), 2522–2534.