Social Collaborative Mutual Learning for Item Recommendation

TIANYU ZHU, Tsinghua University
GUANNAN LIU, Beihang University
GUOQING CHEN, Tsinghua University

Recommender Systems (RSs) provide users with item choices based on their preferences reflected in past interactions and become important tools to alleviate the information overload problem for users. However, in real-world scenarios, the user–item interaction matrix is generally sparse, leading to the poor performance of recommendation methods. To cope with this problem, social information is introduced into these methods in several ways, such as regularization, ensemble, and sampling. However, these strategies to use social information have their limitations. The regularization and ensemble strategies may suffer from the over-smoothing problem, while the sampling-based strategy may be affected by the overfitting problem. To overcome the limitations of the previous efforts, a novel social recommendation model, namely, Social Collaborative Mutual Learning (SCML), is proposed in this article. SCML combines the item-based CF model with the social CF model by two well-designed mutual regularization strategies. The embedding-level mutual regularization forces the user representations in two models to be close, and the output-level mutual regularization matches the distributions of the predictions in two models. Extensive experiments on three public datasets show that SCML significantly outperforms the baseline methods and the proposed mutual regularization strategies can embrace the advantages of the item-based CF model and the social CF model to improve the recommendation performance.

CCS Concepts: • **Information systems** → *Recommender systems*; • **Human-centered computing** → *Social recommendation*;

Additional Key Words and Phrases: Recommender systems, collaborative filtering, social network, mutual learning

ACM Reference format:
Tianyu Zhu, Guannan Liu, and Guoqing Chen. 2020. Social Collaborative Mutual Learning for Item Recommendation. *ACM Trans. Knowl. Discov. Data* 14, 4, Article 43 (May 2020), 19 pages.
https://doi.org/10.1145/3387162

The work was partly supported by the National Natural Science Foundation of China (71490724/71701007) and the MOE Project of Key Research Institute of Humanities and Social Sciences at Universities (17JJD630006).
Authors' addresses: T. Zhu and G. Chen, Department of Management Science and Engineering, Tsinghua University, Beijing 100084, China; emails: {zhuty.16, chengq}@sem.tsinghua.edu.cn; G. Liu (corresponding author), School of Economics and Management, Beihang University, Beijing 100191, China; email: liugn@buaa.edu.cn.

1556-4681/2020/05-ART43 $15.00
https://doi.org/10.1145/3387162

1 INTRODUCTION

Recommender systems (RSs) have nowadays become universally applicable tools and played an important role in e-commerce recently [1, 14, 16, 17, 25]. Given sets of users and items, RSs are designed to recommend items to users according to the purchase history or past ratings. For both customers and sellers, RSs can not only provide customers with items that they might prefer and help them save time when looking for items but also provide sellers with personalized services to increase sales.

RSs provide item choices according to users' preferences, which are often expressed explicitly in the form of ratings or implicitly in the form of clicks, purchases, or reviews. *Collaborative filtering* (CF) has been widely adopted in RSs, where users with similar preferences tend to obtain similar item recommendations [1, 14, 16, 25]. In recent years, item-based CF methods have been widely used in various e-commerce platforms [16]. For these methods, the prediction score of a user to an item is calculated by the similarity of this item and the items that the user has interacted with. However, in real-world applications, the data sparsity problem emerges since it is highly challenging for these methods to infer users' preferences with a few past interactions.

One potential way of coping with this problem is to explore available social networks formed in online review or electronic commerce websites. Online product review websites such as Ciao.com and Epinions.com are the platforms for users to connect, and share their views and/or experiences on their bought items. According to the social influence theory [5], people can be affected by others' emotions, opinions, and behaviors around them, which also applies to the online shopping context. In this regard, in addition to capturing the matching degree between users and items, it is also deemed meaningful to further leverage social information for capturing the preference similarities that people have with their friends.

Recently, many efforts have been made to improve recommendation accuracy by exploiting the users' social relations [11, 19, 20, 30, 37]. These social recommendation methods could be approximately classified into three categories: regularization, ensemble, and sampling. The regularization methods assume that the preferences between connected users should be similar. Therefore, these methods incorporate a regularization term into the loss function, which pulls the embedding vectors of connected users closer [11, 19, 20]. However, these methods may suffer from the over-smoothing problem, since such strategies may hamper the embedding vectors from capturing the differences of preference between connected users. For the ensemble methods [3, 18], the latent vector of a user is represented as a combination of his embedding and his social relations' embedding. Such a strategy may also cause the over-smoothing problem since it is unable to cope with the situation where users with the same social relations have interactions with different items. The sampling-based methods assume that users tend to assign higher ranks to items that their friends prefer [33, 40]. However, this kind of strategies may suffer from the overfitting problem, since the items that a user's friends have interacted with are generally sparse.

To cope with the problems in previous efforts, we propose a novel social recommendation model, namely, *Social Collaborative Mutual Learning* (SCML). The idea of SCML is to combine the item-based CF model with the social CF model by two well-designed mutual regularization strategies so that the two models can complement each other for their advantages. Specifically, SCML is composed of three parts. First, an item-based CF model, Mult-DAE [15], is used to learn user preferences and make predictions according to the items that the user has interacted with. Second, a social CF model, Mult-STE, is proposed to capture the user and his social relations' preferences and predict his interactions. Third, two kinds of mutual regularization strategies are proposed to combine the two models. The embedding-level regularization pulls the user representations in two models closer to let the two models regularize each other on the embedding layer, while the output-level regularization matches the distributions of the predictions for a user in two models,

which performs regularization on the prediction layer. Experiments on three public datasets are conducted to verify the superiority of SCML to the baseline methods. The results also show that the proposed strategies can embrace the advantages of the item-based CF model and the social CF model to improve the recommendation performance.

The rest of this article is organized as follows. Section 2 discusses the related work, Section 3 introduces the notations and formulates the problem, Section 4 presents the proposed model in detail, Section 5 discusses its relations to existing methods and analyzes its time complexity, and Section 6 evaluates its experimental performance, along with its robustness and scalability. Finally, Section 7 concludes the work with possible future studies.

2 RELATED WORK

This section discusses related work on recommendation methods with implicit feedback, social recommendation methods, deep learning-based recommendation methods, and knowledge distillation.

2.1 Recommendation with Implicit Feedback

User feedback reflects user preferences and is frequently observed in real-life contexts in the forms of ratings, reviews, clicks, and so on. Implicit feedback (e.g., clicks and reviews) indirectly expresses how a user prefers an item and is often formulated based on whether a user has an interaction with an item. Generally, such user-item interactions are binary in nature (e.g., 1 if observed and 0 if otherwise). Notably, a value of 0 for user feedback does not necessarily imply feedback that is negative or less favorable, but a possibility that the user is unaware of the existence of the item.

Recommendation with implicit feedback is often treated as a learning-to-rank problem, commonly known as one-class CF or personalized ranking. In Weighted Matrix Factorization (WMF) [10], all the missing feedbacks are marked as negative instances and different weights are given to positive and negative instances, respectively. In Bayesian Personalized Ranking (BPR) [22], negative instances are sampled from the missing feedback and a pairwise model is learned to rank the positive items before the negative ones. In Neural Collaborative Filtering (NCF) [8], the problem is coped with in a binary classification manner, where the negative instances are uniformly sampled from unobserved interactions and the sampling ratio is controlled w.r.t. the number of observed interactions.

2.2 Social Recommendation

Social recommendation has been studied since 1997 [12] and has attracted increasing attention with the growing popularity of social media. Here, social recommendation is referred to as the recommendation with social relations as an additional input [30]. Since users' preferences are likely to be influenced by their social relations, social RSs can be designed by leveraging these relations [3].

Social recommendation methods could be summarized in the following three categories: regularization, ensemble, and sampling. The regularization methods assume that users' preferences should be similar to those of their social relations. In SoRec [19], the user embedding matrix is learned by jointly factorizing the rating matrix and social adjacency matrix. In SocialMF [11], the preference of a user is forced to be close to the average preference of the user's social network. In SoReg [20] and NSCR [32], the preference closeness of two connected users is controlled by their similarity based on their previous ratings. The ensemble methods assume that users and their social relations should have similar ratings on items. In Social Trust Ensemble (STE) [18], a missing rating for a user is predicted as a linear combination of ratings of the user and his social relations. In SAMN [2], an attention-based memory module is employed to build the user-friend specific

relation vector and a friend level attention mechanism is designed to measure the social influence strength. In ARSE [28], the dynamic and static interests of users are modeled by two attention networks to generate user representations with social influence for temporal recommendation. In CNSR [34], the social correlation embedding generated by an autoencoder is incorporated into the user latent embedding to strengthen user representation for item recommendation. The sampling-based methods assume that users' social relations will influence their preference ranking for items. In SBPR [40], social relations are incorporated into the BPR model by assuming that users tend to assign higher ranks to items that their friends prefer. In SamWalker [4], social information is leveraged to guide the sampling process and different weights are specified to different data according to users' social relations.

2.3 Recommendation via Deep Learning

Recent studies demonstrate the effectiveness of deep learning in coping with recommendation tasks. The methods of deep learning-based recommendation could be divided into the following two categories: the methods based on matching function learning and the methods based on representation learning. The methods based on matching function learning use deep models to learn complex patterns of user-item interactions. In NCF [8], the matching function is learned from deep neural networks instead of inner product in Matrix Factorization (MF). Then, NeuMF is proposed as a combined framework that concatenates the generalized MF with a multi-layer perceptron (MLP). In ConvNCF [6], outer products and two-dimensional convolution layers are exploited for learning joint representations of user-item pairs. In LRML [38], an attention-based memory-augmented neural architecture is proposed to model the relationship between users and items in metric space using latent relation vectors.

The methods based on representation learning leverage deep models to learn the representations of users or items. In RBM-CF [24], Restricted Boltzmann Machines (RBMs) are used for collaborative filtering, which is one of the first attempts that generate recommendations via deep learning. In AutoRec [26], a vanilla autoencoder is used for rating prediction. In CDAE [35], a stacked denoising autoencoder (DAE) is used to learn the latent representations of corrupted user-item preferences that can best reconstruct the observed interactions. In DMF [36], the row vectors and column vectors of the rating matrix are used to learn the user and item representation vectors respectively and the cosine similarity is used as the matching function. In Mult-VAE [15], a generative model is introduced with a multinomial likelihood function parameterized by neural networks, along with a variant of variational autoencoder developed for collaborative filtering on implicit feedback data.

2.4 Knowledge Distillation and Mutual Learning

Knowledge distillation is a strategy for model compression, which has been used in computer vision and other areas [9]. First, a larger teacher model is trained to obtain a smooth label distribution. Then a smaller student model is trained by optimizing the loss from two parts: the ground-truth label distribution and the smoothed pseudo label distribution generated by the teacher model. In this way, the more information of label distribution is used as additional supervision to train the smaller student model, improving its generalization ability on test instances. FitNet [23] extends this idea by using both the outputs and the intermediate representations from the teacher model to guide the training process of the student model. Since the hidden layer of the student model is generally smaller than that of the teacher model, an additional transformation is proposed to map the student hidden layer to the prediction of the teacher hidden layer, which allows the student model to run faster and obtain better performance. Ranking Distillation [31] introduces this idea into ranking models. It trains a smaller student model to learn to rank items based on the training data and the result of a larger teacher model to achieve a similar ranking performance while

Table 1. Notations

Notation	Description
U, I	Set of users/items
$\mathbf{R}$	Observed user-item interaction matrix
$\mathbf{r}_u$	Observed user-item interaction vector for user u
$\hat{\mathbf{r}}_u^{ICF}, \hat{\mathbf{r}}_u^{SCF}$	Predicted user-item interaction vector for user u by item-based/social CF model
$\tilde{\mathbf{r}}_u^{ICF}, \tilde{\mathbf{r}}_u^{SCF}$	Normalized predicted user-item interaction vector for user u by item-based/social CF model
$\mathbf{A}$	User adjacency matrix
$\mathbf{t}_u$	Adjacency vector for user u with self-loop
$\mathbf{p}_u$	Preference embedding vector for user u
$\mathbf{s}_u$	Social embedding vector for user u
$\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3$	Item embedding matrix
$\mathbf{V}$	User embedding matrix
$\mathbf{b}_1, \mathbf{b}_3$	Embedding bias vector
$\mathbf{b}_2, \mathbf{b}_4$	Item bias vector
λ_{emb}	Coefficient for embedding-level mutual regularization
λ_{out}	Coefficient for output-level mutual regularization
λ_{reg}	Coefficient for l_2 regularization
d	Embedding size

making the online inference more efficient. Different from distillation strategies, mutual learning is a strategy that several models learn collaboratively and teach each other during the training process [39]. It incorporates the Kullback Leibler (KL) divergence of the models' predictions into the loss function to improve the generalization performance of models for classification problems.

3 PRELIMINARIES

Prior to proposing our model, this section first introduces the notations and formulates the problem in this article.

3.1 Problem Formulation

The recommendation task considered in this article takes implicit feedback as the training data. Let U and I be the sets of users and of items, respectively. In a social recommendation scenario, two kinds of information are available: the binary interaction matrix $\mathbf{R} \in \mathbb{R}^{|U|\times|I|}$ where $r_{ui} = 1$ if user u has an interaction with item i and $r_{ui} = 0$ otherwise, and the binary adjacency matrix of user social network $\mathbf{A} \in \mathbb{R}^{|U|\times|U|}$ where $a_{uv} = 1$ if user u and user v have an edge in the social network and $a_{uv} = 0$ otherwise. Other notations used throughout this article are provided in Table 1.

Given the user-item interaction matrix $\mathbf{R}$ and the user adjacency matrix $\mathbf{A}$, the task of social recommendation in this article is to predict whether user u has a potential interest in item i that he has not interacted with. Specifically, the goal here is to recommend a ranked list of items to each user that he is most likely interested in according to the prediction score.

4 THE PROPOSED MODEL

This section introduces the proposed SCML model. First, the whole framework of SCML is illustrated. Next, the details of SCML are described, including the item-based CF model, the social CF

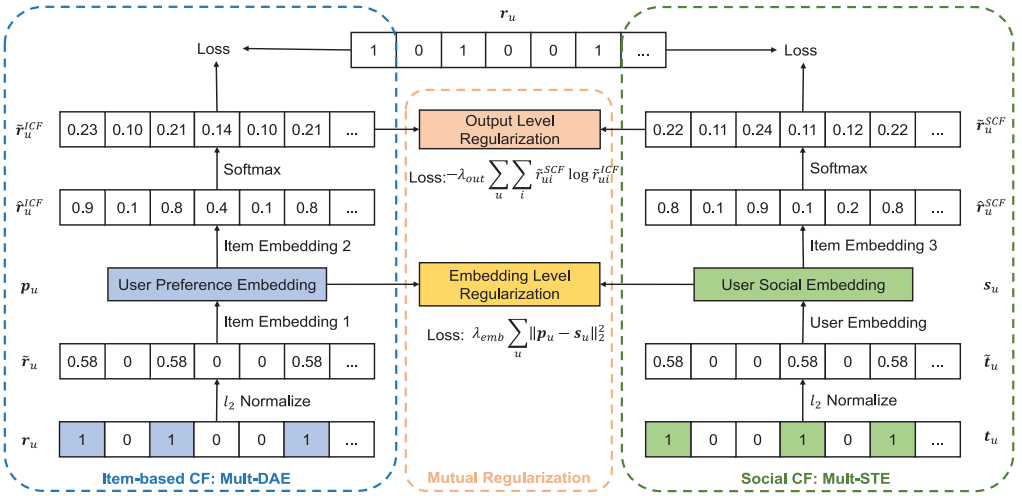


Fig. 1. The framework of SCML.

model, and the two mutual learning strategies. Then, the optimization and recommendation processes for SCML are introduced. Finally, the relations of SCML to existing methods are discussed.

4.1 The Framework

The proposed SCML model is to jointly optimize the item-based CF model and the social CF model with mutual regularization for item recommendation, with the framework as shown in Figure 1. For the left part, an item-based CF model is used, which is a two-layer DAE, to map the users' interaction vectors into a latent space and reconstruct the vectors from the latent representations. On the right, a social CF model is proposed, which maps users' adjacency vectors in the social network into a latent space to represent the preference of users' social group and predict their preferences by the latent representations. In the middle is the mutual regularization part. The following two kinds of mutual regularization are proposed: the embedding-level regularization that is applied to the user representation layers and the output-level regularization that is performed on the prediction layers. The two mutual regularization strategies allow the two models to complement each other for their own advantages, thus improve the recommendation performance. In the following subsections, the SCML model is elaborated in detail for different parts.

4.2 Item-based Collaborative Filtering

Following [15], a DAE is used to build an item-based CF model, namely, Mult-DAE. In this model, a binary multi-hot encoding vector $\mathbf{r}_u$, which denotes user u's interactions on the entire item set, is used as input. To avoid overfitting, noise is added to $\mathbf{r}_u$ by dropout [27] during the training procedure. Moreover, l_2 normalization is applied to the input layer.

$$\tilde{\mathbf{r}}_u = \frac{\text{dropout}(\mathbf{r}_u)}{||\text{dropout}(\mathbf{r}_u)||_2}. \tag{1}$$

For simplification, a two-layer autoencoder is used here. Above the input layer is the embedding layer, which is a fully-connected layer that projects the sparse input vector to a dense vector. The obtained embedding vector $\mathbf{p}_u$ can be seen as the user preference embedding, which is the weighted sum of the embedding vectors of items that user u has interacted with.

$$\mathbf{p}_u = \tanh(\tilde{\mathbf{r}}_u\mathbf{W}_1 + \mathbf{b}_1), \tag{2}$$

where $\mathbf{W}_1 \in \mathbb{R}^{|I| \times d}$ is an item embedding matrix and $\mathbf{b}_1 \in \mathbb{R}^d$ is a bias vector for item embedding. The user preference embedding $\mathbf{p}_u$ is then transformed via a fully-connected layer to produce the relevance scores over the entire set of items. This can be seen as the inner product of the user preference embedding vector and each embedding vector of items, with the addition of item bias

$$\hat{\mathbf{r}}_u^{ICF} = \mathbf{p}_u \mathbf{W}_2 + \mathbf{b}_2, \tag{3}$$

where $\mathbf{W}_2 \in \mathbb{R}^{|I| \times d}$ is another item embedding matrix and $\mathbf{b}_2 \in \mathbb{R}^{|I|}$ is an item bias vector. The output is then normalized via a softmax function to produce a probability distribution over the whole item set

$$\tilde{\mathbf{r}}_u^{ICF} = \text{softmax}(\hat{\mathbf{r}}_u^{\text{ICF}}) = \frac{\exp(\hat{\mathbf{r}}_u^{ICF})}{\sum_j \exp(\hat{r}_{uj}^{ICF})}. \tag{4}$$

Then the model maximizes the multinomial likelihood for optimization, which is equal to the cross-entropy loss for multi-class classification.

$$L_{ICF} = -\sum_u \sum_i r_{ui} \log \tilde{r}_{ui}^{ICF}. \tag{5}$$

4.3 Social Collaborative Filtering

A variant of STE model is designed for social collaborative filtering, namely, Mult-STE. In this model, a binary multi-hot encoding vector $\mathbf{t}_u$, which is user u's adjacency vector in the social network with self-loops, is used as input. The input vector denotes the IDs of user u and his friends. Similar to Mult-DAE, the input vector is l_2-normalized before fed into the next layer

$$\mathbf{T} = \mathbf{A} + \mathbf{I}, \tag{6}$$

$$\tilde{\mathbf{t}}_u = \frac{\mathbf{t}_u}{||\mathbf{t}_u||_2}. \tag{7}$$

Then a fully-connected layer is used above the input layer to produce the user social embedding, which is the weighted sum of the embedding vectors of user u and his friends. This can be seen as a linear combination of user u's latent preference and his friends'

$$\mathbf{s}_u = \tanh(\tilde{\mathbf{t}}_u \mathbf{V} + \mathbf{b}_3), \tag{8}$$

where $\mathbf{V} \in \mathbb{R}^{|U| \times d}$ is a user embedding matrix and $\mathbf{b}_3 \in \mathbb{R}^d$ is a bias vector for user embedding. Similar to Mult-DAE, the user social embedding $\mathbf{s}_u$ is then transformed via a fully-connected layer to produce the relevance scores over the entire set of items. The output is normalized via a softmax function to produce a probability distribution over the whole item set

$$\hat{\mathbf{r}}_u^{SCF} = \mathbf{s}_u \mathbf{W}_3 + \mathbf{b}_4, \tag{9}$$

$$\tilde{\mathbf{r}}_u^{SCF} = \text{softmax}(\hat{\mathbf{r}}_u^{\text{SCF}}) = \frac{\exp(\hat{\mathbf{r}}_u^{SCF})}{\sum_j \exp(\hat{r}_{uj}^{SCF})}, \tag{10}$$

where $\mathbf{W}_3 \in \mathbb{R}^{|I| \times d}$ is an item embedding matrix and $\mathbf{b}_4 \in \mathbb{R}^{|I|}$ is an item bias vector. Finally, the model maximizes the multinomial likelihood for optimization.

$$L_{SCF} = -\sum_u \sum_i r_{ui} \log \tilde{r}_{ui}^{SCF}. \tag{11}$$

4.4 Mutual Regularization

To combine the item-based CF model Mult-DAE and the social CF model Mult-STE, the following two kinds of mutual-learning strategies are designed: the embedding-level mutual regularization and the output-level mutual regularization.

4.4.1 Embedding-Level Mutual Regularization. The idea of the embedding-level mutual regularization is that the user representations in the two models should be close. On one hand, in Mult-DAE, the user representation vector is denoted as user preference embedding $\mathbf{p}_u$, which is the weighted sum of item embedding that user u has interacted with. It reflects users' latent preference from the viewpoint of items. On the other hand, in Mult-STE, the user representation vector is denoted as user social embedding $\mathbf{s}_u$, which is the weighted sum of the embedding vectors of user u and his friends. It reflects users' latent preferences from the viewpoint of their social relations. To combine users' latent preferences from the two viewpoints, the embedding-level mutual regularization strategy is proposed. Here the square of Euclidean distance between user preference embedding and user social embedding is used as the mutual learning loss.

$$L_{emb} = \sum_{u \in U} ||\mathbf{p}_u - \mathbf{s}_u||_2^2. \tag{12}$$

4.4.2 Output-Level Mutual Regularization. The idea of the output-level mutual regularization is that the distributions of predicted preference scores for a user in two models should be consistent. For Mult-DAE, the predicted preference scores are computed as the similarity between the candidate items and the items that the user has interacted with. For Mult-STE, the preference scores seem to come from group decisions, since the predictions are made by the members of the user's social group. To combine the individual and group level item preferences for improving recommendation accuracy, the output-level mutual regularization strategy is proposed. Since the predicted preference scores over the entire item set is a probability distribution after softmax transformation, here the KL divergence from $\tilde{\mathbf{r}}_u^{ICF}$ to $\tilde{\mathbf{r}}_u^{SCF}$ is adopted for mutual regularization.

$$\begin{aligned} D_{KL} &= \sum_{u \in U} \mathrm{KL}\left(\tilde{\mathbf{r}}_u^{SCF} || \tilde{\mathbf{r}}_u^{ICF}\right) \\ &= \sum_{u \in U} \sum_{i \in I} \tilde{r}_{ui}^{SCF} \log \left(\frac{\tilde{r}_{ui}^{SCF}}{\tilde{r}_{ui}^{ICF}} \right) \\ &= \sum_{u \in U} \sum_{i \in I} \tilde{r}_{ui}^{SCF} \log \tilde{r}_{ui}^{SCF} - \tilde{r}_{ui}^{SCF} \log \tilde{r}_{ui}^{ICF}. \end{aligned} \tag{13}$$

Since we focus on the mutual regularization of two models, the cross-entropy in KL divergence is chosen as the mutual learning loss.

$$L_{out} = -\sum_{u \in U} \sum_{i \in I} \tilde{r}_{ui}^{SCF} \log \tilde{r}_{ui}^{ICF}. \tag{14}$$

In this way, the two models also learn from each other to improve their generalization performance.

4.5 Optimization

The overall loss function for SCML is as follows:

$$L = L_{ICF} + \alpha L_{SCF} + \lambda_{emb} L_{emb} + \lambda_{out} L_{out} + \lambda_{reg} L_{reg}, \tag{15}$$

where α is a scalar that balances the losses of two models, which is set to 1 as default. λ_{emb} and λ_{out} are the coefficients to control the embedding and output-level mutual regularization and L_{reg}

is an l_2 regularization term to prevent overfitting, which is defined as follows:

$$L_{reg} = \frac{1}{2}\left(\left\|\mathbf{W}_1\right\|_F^2 + \left\|\mathbf{W}_2\right\|_F^2 + \left\|\mathbf{W}_3\right\|_F^2 + \left\|\mathbf{V}\right\|_F^2\right). \tag{16}$$

To optimize the objective function of SCML, mini-batch *Adaptive Moment Estimation* (Adam) [13] is used, which is a variant of *Stochastic Gradient Descent* (SGD) with adaptive moment estimation. Specifically, in each epoch, all the users are shuffled first. Then, mini-batch of users is sampled and their interaction vectors are put into the model. The gradient of the loss function is calculated with respect to each trainable parameter. The trainable parameters are updated according to the Adam algorithm until convergence.

5 DISCUSSION

This section discusses the relations of SCML to knowledge distillation and mutual-learning methods and analyzes the time complexity of SCML.

5.1 Relations to Existing Methods

5.1.1 Knowledge Distillation. For knowledge distillation [9], a smaller student model is trained based on both the ground-truth label distribution and the smoothed label distribution generated by the pre-trained larger teacher model. In this way, the student model improves its generalization ability for test instances. Different from knowledge distillation in [9] or ranking distillation in [31], SCML does not use a pre-trained teacher model to guide the learning process of the student model. Instead, it allows an ensemble of two models from different perspectives to teach each other for mutual regularization. In this way, the item-based CF model Mult-DAE can learn the information from the social CF model Mult-STE to improve its performance for cold-start users. Meanwhile, Mult-STE can overcome the over-smoothing problem and improve its performance for warm-start users by mutual learning with Mult-DAE. Therefore, SCML can combine the advantages of the two models to improve the recommendation performance.

5.1.2 Mutual Learning. The original mutual-learning strategy allows several models to learn collaboratively and teach each other during the training process [39]. The KL divergence of the models' predictions is added to the loss function to improve the generalization performance of the models. Different from mutual learning in [39], SCML applies the mutual regularization strategies to a ranking task. The two models in SCML generate predictions from different perspectives to complement each other for their advantages. In addition, the embedding-level mutual regularization strategy is proposed in SCML, which performs on different layers with the strategy in [39].

5.2 Time Complexity Analysis

Here the time complexity of the training process of SCML is analyzed. For Mult-DAE, the time complexity for a training epoch is $O(|U||I|d)$, where $|U|$ and $|I|$ denotes the number of users and items respectively and d denotes the embedding size. Compared to Mult-DAE, the additional cost of SCML comes from Mult-STE and the two kinds of mutual regularization. The time complexity of Mult-STE for a training epoch is $O(|U|^2d + |U||I|d)$. For the embedding-level mutual regularization, the time complexity is $O(|U|d)$. And the output-level mutual regularization takes time $O(|U||I|)$. Therefore, the overall time complexity of SCML for a training epoch is $O(|U|^2d + |U||I|d)$. Note that for the cases where $|U|$ is at the same level as $|I|$, which are common in real applications including the data scenarios and related contexts of our experiments, the time complexity of SCML is at the same level as that of Mult-DAE.

Table 2. Dataset Statistics

Statistics	Ciao	Epinions	Yelp
#users	5,945	20,608	37,579
#items	11,211	23,585	42,444
#ratings	145,401	454,022	503,903
#connections	96,917	351,485	139,994
rating density	0.218%	0.093%	0.032%
connection density	0.274%	0.083%	0.010%

6 EXPERIMENTS

In this section, data experiments were conducted to verify the superiority of SCML to other related state-of-the-art recommendation methods on three real-world datasets. The codes of SCML are published on GitHub.[1] The experiments were designed to answer the following questions:

RQ1. Does SCML outperform state-of-the-art recommendation methods?
RQ2. How does mutual learning affect the model's performance?
RQ3. How do hyperparameters influence the performance of SCML?
RQ4. How does SCML perform for users with different scales of interactions?

6.1 Datasets

The three experimental datasets are publicly accessible, namely, Ciao, Epinions, and Yelp.

Ciao.[2] This public dataset was crawled from the popular product review site Ciao in the month of May 2011 [29]. It contains trust relationships among users as well as their ratings.

Epinions.[3] This dataset was collected in a 5-week crawl (November/December 2003) from the Epinions.com Web site [21]. It consists of user-user trust relationships and user-item ratings.

Yelp Challenge 2019.[4] This dataset was a subset of Yelp's businesses, reviews, and user data for academic use. The user-business rating data and user-user friendship data were used in the experiments.

The original data of the three datasets were large but highly sparse. Thus, the datasets were filtered in the way that retained only items and users with at least five records.[5] For the first two datasets, the presence of a review was treated as implicit feedback. For the Yelp dataset, a rating over three was treated as implicit feedback. The statistics of the three datasets are summarized in Table 2.

6.2 Experimental Settings

Evaluation Protocols. To evaluate the performance of item recommendation, the leave-one-out evaluation was adopted, which has been widely used in literature [8, 22]. The data were split into training/validation/test sets by randomly selecting for each user an item to be used for validation and another for testing. All remaining data were used for training. Since it was too time-consuming to rank all the items for every user during the evaluation, the common strategy was employed that 99 items that had not been interacted by the user were randomly sampled and the test item was

[1] http://github.com/zhuty16/SCML.
[2] http://www.cse.msu.edu/~tangjili/trust.html.
[3] http://www.trustlet.org/downloaded_epinions.html.
[4] http://www.yelp.com/dataset/challenge.
[5] Since the original Yelp dataset is too large to process, 300,000 users and their ratings were randomly sampled before filtering.

ranked among these 99 items [7, 8]. The performance of a ranked list was judged by Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG). Without special mention, the ranked list was truncated at 10 for both metrics. As such, the HR intuitively measured whether the test item was presented on the top-10 list, and the NDCG accounted for the position of the hit by assigning higher scores to hits at top ranks. Both metrics were calculated for each test user and the average scores were reported. Each experiment was conducted for 10 times and the average results were reported.

Baselines. To show the effectiveness of SCML, two groups of recommendation methods were chosen as baselines. The first group contains three general recommendation methods that only consider users' implicit feedback as input:

- **Most Popular.** This is a non-personalized method simply ranking items by the number of interactions.
- **BPR[22].** This is an MF-based method with a pairwise ranking loss, which is tailored to learn from implicit feedback.
- **NeuMF[8].** This is a state-of-the-art deep model for item recommendation, which combines the hidden layers of MF and MLP to learn the user-item interaction function.

The second group includes five social recommendation methods that use users' social information to improve the recommendation performance:

- **SoRec[19].** This method simultaneously factorizes the user-item rating matrix and the user-user adjacency matrix for rating prediction. Here the square loss was modified as the BPR loss to suit the item recommendation task.
- **SoReg[20].** This method is based on MF and controls the preference closeness of connected users by their rating similarities. Here the square loss was modified into the BPR loss to suit the item recommendation task.
- **SBPR[40].** This method is an extension of BPR that considers social relations in the learning process, assuming that users tend to assign higher ranks to the items that their friends prefer.
- **NeuMF-SoReg.** Like SoReg, social regularization on embedding vectors of connected users is added into the original NeuMF model to realize a deep social recommendation method.
- **SAMN[2]** This is a deep social recommendation method that employs the attention-based memory module to construct the user-friend specific relation vector and designs the friend-level attention to measure the social influence strength among users' friends for item recommendation.

Note that here we denote the Mult-DAE and Mult-STE models with the proposed two mutual regularization strategies as Mult-DAE-SCML and Mult-STE-SCML.

Parameter Settings. All the methods were implemented based on TensorFlow.[6] For model-based methods, the embedding size d was set to 64 as default and all the parameters were initialized by the random normal initializer (with a mean of 0 and standard deviation of 0.01)[8] and were optimized with mini-batch Adam, in the batch size of 256. Other hyperparameters were tuned on the validation sets by grid search: the learning rate was chosen from $\{1e\text{-}4, 5e\text{-}4, 1e\text{-}3, 5e\text{-}3\}$, the l_2 regularization coefficient was chosen from $\{0, 1e\text{-}5, 1e\text{-}4, 1e\text{-}3\}$, and the social regularization coefficient for SoRec, SoReg or NeuMF-SoReg was chosen from $\{0.001, 0.01, 0.1, 1, 10\}$. Following [8], a tower structure with embedding size of $128 \rightarrow 64 \rightarrow 32 \rightarrow 16$ was adopted for NeuMF and NeuMF-SoReg.

[6] http://www.tensorflow.org/.

Table 3. Performance Comparison on Three Datasets

	Ciao		Epinions		Yelp	
Method	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
Most Popular	0.3963	0.2486	0.4859	0.3014	0.4458	0.2627
BPR	0.5463	0.3565	0.6869	0.4730	0.8335	0.5672
NeuMF	0.5103	0.3244	0.6266	0.4145	0.7483	0.4915
SoRec	0.5516	0.3591	0.6924	0.4759	0.8340	0.5670
SoReg	0.5807	0.3823	0.6959	0.4773	0.8350	0.5700
SBPR	0.4866	0.2915	0.6377	0.4219	0.8125	0.5449
NeuMF-SoReg	0.5369	0.3380	0.6361	0.4202	0.7614	0.5075
SAMN	0.5574	0.3644	0.6788	0.4604	0.8090	0.5474
Mult-DAE	0.5988	0.4003	0.6979	0.4844	0.8365	0.5868
Mult-STE	0.5623	0.3668	0.6583	0.4475	0.8083	0.5545
Mult-DAE-SCML	**0.6121***	**0.4063***	**0.7079***	**0.4921***	**0.8379**	**0.5934***
Mult-STE-SCML	0.5859	0.3856	0.6788	0.4682	0.8305	0.5810

Best results are in boldface and second best are underlined. * means statistically significant improvements compared to the best baseline by unpaired two-sample t-test with $p = 0.01$.

6.3 Performance Comparison (RQ1)

Experimental results in terms of *HR*@10 and *NDCG*@10 on three datasets are shown in Table 3. First, model-based methods significantly outperformed the simplest most popular method. However, deep learning-based methods (i.e., NeuMF, NeuMF-SoReg, and SAMN) did not obtain better results than shallow methods. Second, SoRec and SoReg performed better than BPR, which showed that social information could help improve the recommendation accuracy. For other social recommendation methods, NeuMF-SoReg outperformed NeuMF by adding social regularization, while SBPR did not achieve better performances than BPR. A possible explanation might be that the way of using social information in SBPR is easy to cause the overfitting problem. Third, Mult-DAE obtained comparable results with SoReg, indicating the advantages of global optimization over pairwise sampling optimization. In addition, Mult-DAE performed better than Mult-STE, since users' feedback in these datasets is not too sparse and the item-based model could better capture users' preferences. Moreover, the proposed mutual regularization strategies led to consistent improvements for Mult-DAE and Mult-STE on three datasets, and Mult-DAE with mutual regularization (Mult-DAE-SCML) outperformed all baseline methods significantly.

Experiments using different proportions of the training data were also conducted to validate the effectiveness of SCML with different data sparsity levels. Here 40% and 70% data of the original training set in Ciao and Epinions were sampled randomly and the validation/test set was kept unchanged. The results in Table 4 are worse than those in Table 3 due to the insufficient training data. In this case, the proposed mutual regularization strategies improve the performances of Mult-DAE and Mult-STE more significantly, since the social information plays an essential role to help predict users' preferences when their feedback is highly sparse.

Table 5 shows the training time per iteration of Mult-DAE, Mult-STE, and SCML. A training iteration is defined as training a mini-batch of users for their interactions on the entire item set. The running environment was a server with Intel(R) Xeon(R) Gold 5118 CPU @ 2.30 GHz and a GPU with Tesla P100-PCIE-16GB. Note that the running time of SoReg and other baselines are not shown since they are optimized by pairwise ranking loss and are not comparable with Mult-DAE, Mult-STE and SCML that optimized on the entire item set. The results show that SCML took longer time than Mult-DAE (Mult-STE) due to the additional cost of training Mult-STE (Mult-DAE) and

Table 4. Performance Comparison on Four Sparse Datasets

Method	Ciao40		Ciao70	
	HR@10	NDCG@10	HR@10	NDCG@10
Most Popular	0.3830	0.2447	0.3928	0.2474
BPR	0.3997	0.2588	0.4954	0.3225
NeuMF	0.3970	0.2543	0.4585	0.2951
SoRec	0.4234	0.2751	0.4984	0.3233
SoReg	0.4501	0.2888	0.5255	0.3396
SBPR	0.3815	0.2254	0.4601	0.2759
NeuMF-SoReg	0.4294	0.2747	0.4890	0.3106
SAMN	0.4138	0.2670	0.4952	0.3205
Mult-DAE	0.4532	0.2952	0.5457	0.3602
Mult-STE	0.4632	0.3018	0.5230	0.3358
Mult-DAE-SCML	**0.4846***	**0.3168***	**0.5601***	**0.3696***
Mult-STE-SCML	**0.4849***	**0.3177***	0.5437	0.3550
Method	**Epinions40**		**Epinions70**	
	HR@10	NDCG@10	HR@10	NDCG@10
Most Popular	0.4834	0.3012	0.4843	0.3011
BPR	0.5254	0.3451	0.6334	0.4313
NeuMF	0.4844	0.3062	0.5687	0.3672
SoRec	0.5573	0.3654	0.6427	0.4369
SoReg	0.5728	0.3735	0.6551	0.4449
SBPR	0.4872	0.2998	0.5905	0.3862
NeuMF-SoReg	0.5105	0.3245	0.5882	0.3802
SAMN	0.5501	0.3617	0.6439	0.4341
Mult-DAE	0.5818	0.3857	0.6563	0.4484
Mult-STE	0.5757	0.3789	0.6279	0.4209
Mult-DAE-SCML	**0.6021***	**0.4017***	**0.6669***	**0.4580***
Mult-STE-SCML	**0.5974***	**0.3953***	0.6503	0.4393

Best results are in boldface and second best are underlined. * means statistically significant improvements compared to the best baseline by unpaired two-sample t-test with $p = 0.01$.

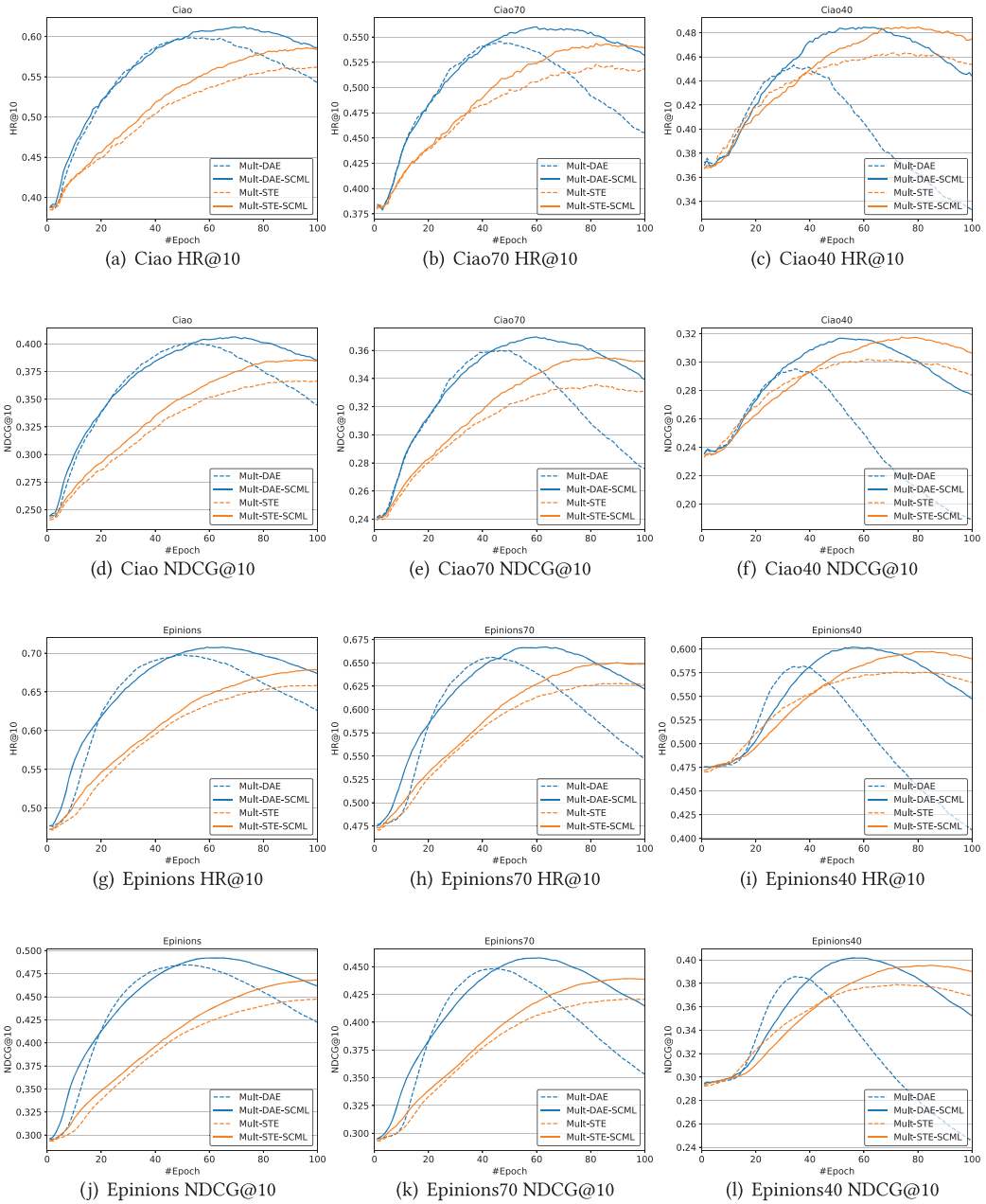
Table 5. Training Time (Seconds) Per Iteration of Mult-DAE, Mult-STE, and SCML Implemented by TensorFlow

Method	Ciao	Epinions	Yelp
Mult-DAE	0.0125s	0.0414s	0.0717s
Mult-STE	0.0175s	0.0726s	0.1240s
SCML	0.0203s	0.0797s	0.1343s

mutual regularization. The additional time cost is quite acceptable for the significant improvements on model performance, which were roughly 0.62, 0.93, and 0.87 times of the training time for Mult-DAE and 0.16, 0.10, and 0.08 times of the training time for Mult-STE on three datasets.

6.4 Effect of Mutual Learning (RQ2)

Figure 2 shows the *HR*@10 and *NDCG*@10 of Mult-DAE and Mult-STE with/without mutual regularization w.r.t the number of epochs on the validation sets of Ciao and Epinions datasets. First,



(a) Ciao HR@10 (b) Ciao70 HR@10 (c) Ciao40 HR@10

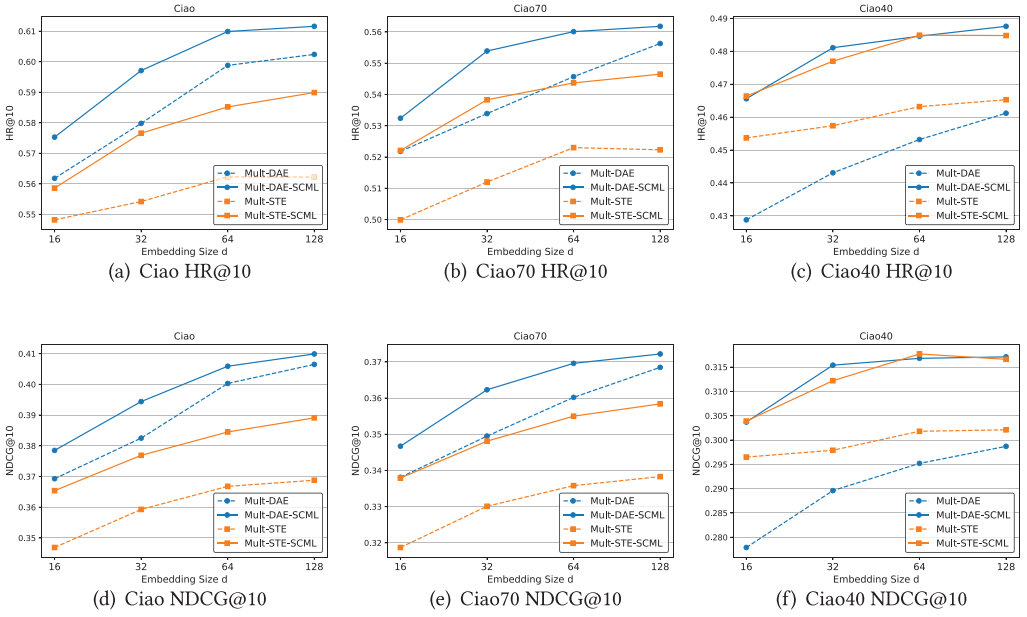
(d) Ciao NDCG@10 (e) Ciao70 NDCG@10 (f) Ciao40 NDCG@10

(g) Epinions HR@10 (h) Epinions70 HR@10 (i) Epinions40 HR@10

(j) Epinions NDCG@10 (k) Epinions70 NDCG@10 (l) Epinions40 NDCG@10

Fig. 2. Model performances w.r.t. number of epochs.

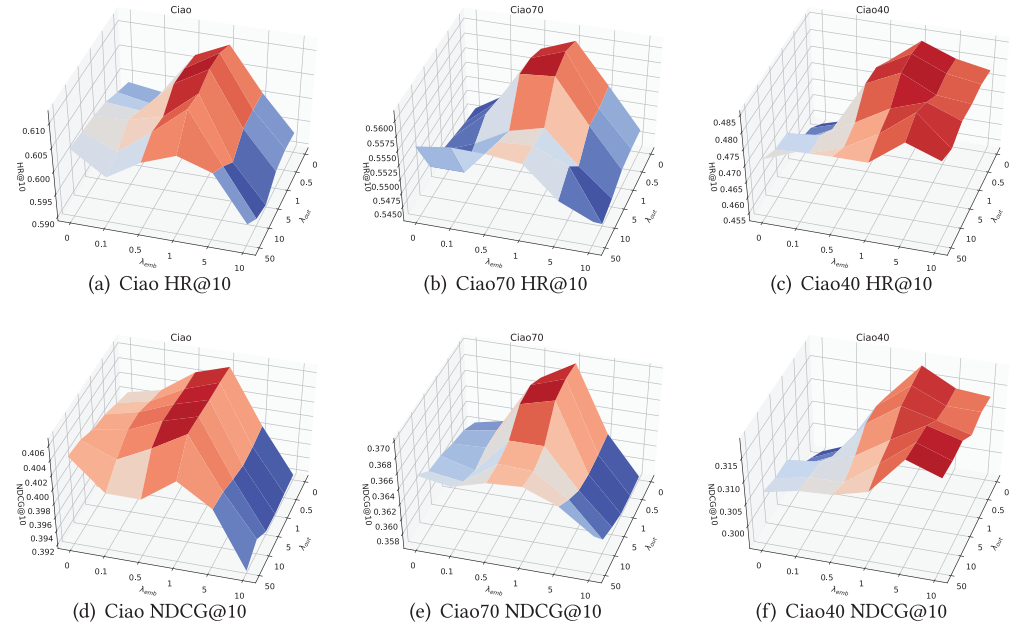
with mutual regularization, Mult-DAE and Mult-STE both obtained better performances on the datasets. Second, when the training data became sparse, mutual regularization led to a more significant improvement in the results. Third, the *HR*@10 and *NDCG*@10 of Mult-DAE and Mult-STE dropped more slowly with mutual learning when overfitting occurred, indicating that mutual regularization could alleviate the overfitting problem, especially when the training data is insufficient.



(a) Ciao HR@10 (b) Ciao70 HR@10 (c) Ciao40 HR@10

(d) Ciao NDCG@10 (e) Ciao70 NDCG@10 (f) Ciao40 NDCG@10

Fig. 3. Model performances w.r.t. embedding size d.



(a) Ciao HR@10 (b) Ciao70 HR@10 (c) Ciao40 HR@10

(d) Ciao NDCG@10 (e) Ciao70 NDCG@10 (f) Ciao40 NDCG@10

Fig. 4. Performances of Mult-DAE-SCML w.r.t. mutual regularization coefficients λ_{emb} and λ_{out}.

6.5 Hyperparameter Sensitivity (RQ3)

SCML involves the following two key hyperparameters: the embedding size d and the mutual regularization coefficient $\lambda_{emb}/\lambda_{out}$. This subsection examines how the different choices of hyperparameters affect the performance of SCML.

Embedding Size d. Figure 3 shows the performances of Mult-DAE and Mult-STE with/without mutual regularization with embedding size in {16, 32, 64, 128}. The trends were generally in consistence with the observations at embedding size 64. First, in most cases, the performances of Mult-DAE and Mult-STE improved with the increase of the embedding size for user/item latent vectors due to the increase of model capacity. Second, the proposed mutual regularization consistently improved the performances of Mult-DAE and Mult-STE with different embedding sizes, showing its significant effect on improving the recommendation accuracy.

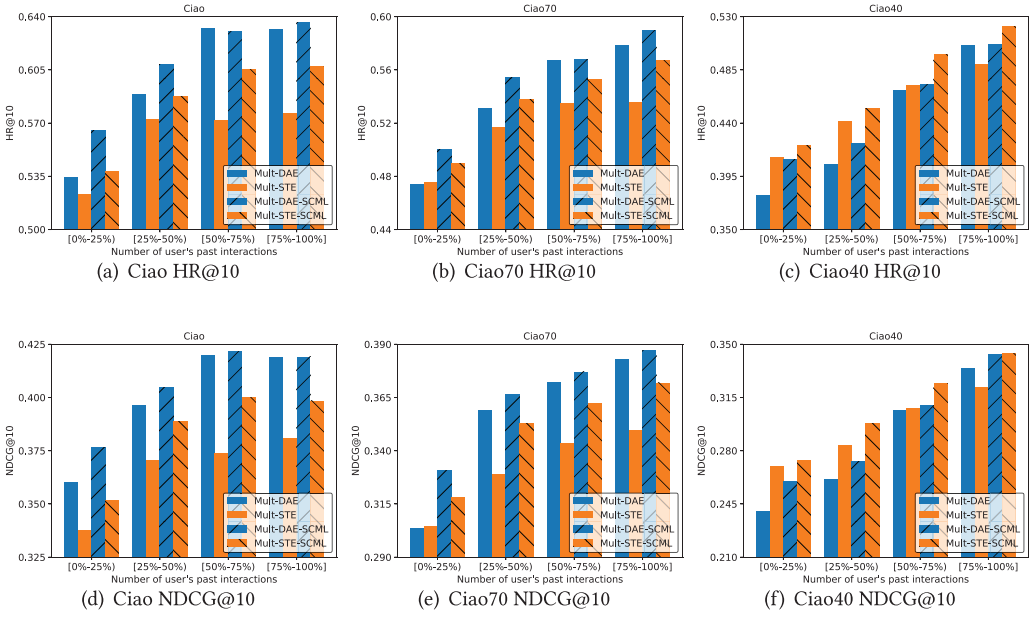
Mutual Regularization Coefficient $\lambda_{emb}/\lambda_{out}$. To examine how the mutual regularization strategies influence the performances of Mult-DAE, the results with different regularization coefficients (i.e., λ_{emb} and λ_{out}) were analyzed, as shown in Figure 4. First, when only the embedding-level mutual regularization was applied, the best results were achieved when $\lambda_{emb} = 1$. While for the output-level mutual regularization, the optimal value of λ_{out} was 50. In addition, the embedding-level mutual regularization appeared to be more efficient than the output-level mutual regularization, especially on the sparse datasets. Second, when the two mutual regularization strategies were jointly used, the performances of Mult-DAE were further improved. Notably, the embedding-level regularization dominated the model performances. This might be because the embedding-level regularization was applied to regularizing the latent representations of users, which affects the results more directly than the output-level regularization that regularizes the preference scores of users across the whole item set. Third, when $\lambda_{emb}/\lambda_{out}$ continued to increase, the model performances went down, especially on the denser datasets. This reveals that moderate mutual regularization could improve model performance while excessive mutual regularization would hamper Mult-DAE from learning users' preferences from their past interactions.

6.6 Performance w.r.t Users with Different Scale of Interactions (RQ4)

In this subsection, experiments were conducted to answer such a question: What kinds of users would benefit from mutual regularization? Users were divided into four groups according to their numbers of past interactions on Ciao datasets and the performances of Mult-DAE and Mult-STE with/without mutual regularization for different groups are shown in Figure 5. First, without mutual regularization, Mult-DAE outperformed Mult-STE in most cases. When users' feedback is highly sparse (for the first group in Ciao70 and the first three groups in Ciao40), Mult-STE obtained better results than Mult-DAE by using social information. Second, for Mult-DAE, users with fewer past interactions (cold-start) obtained larger improvements on recommendation accuracy by mutual regularization. In contrast, for Mult-STE, users with more past interactions (warm-start) benefited more from mutual regularization. Since Mult-DAE suffers from the data sparsity problem and Mult-STE suffers from the over-smoothing problem, the proposed mutual regularization strategies can embrace the advantages of Mult-DAE and Mult-STE for better recommendation performance.

7 CONCLUSION AND FUTURE WORK

This article has proposed a novel social recommendation model, namely, SCML, which combines an item-based CF model with a social CF model by two well-designed mutual regularization strategies. The embedding-level mutual regularization forces the user representations in two models to be close, which performs regularization on the embedding layer; while the output-level mutual regularization matches the distributions of the predictions in two models, which allows the two models to regularize each other on the prediction layer. The proposed mutual regularization strategies can combine the advantages of the two models to improve the recommendation performance. Extensive experiments on three benchmark datasets have revealed the superiority of SCML to competitive baselines.



(a) Ciao HR@10 (b) Ciao70 HR@10 (c) Ciao40 HR@10

(d) Ciao NDCG@10 (e) Ciao70 NDCG@10 (f) Ciao40 NDCG@10

Fig. 5. Model performances w.r.t. users with different scale of interactions.

In SCML, the weight of mutual regularization is set to be equal for all users, which might be unreasonable for users with different numbers of past interactions and friends. Future work will focus on how to design the user-specific weight of mutual regularization for better mutual learning between the two models.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering* 17, 6 (2005), 734–749.

[2] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2019. Social attentional memory network: Modeling aspect- and friend-level differences in recommendation. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*. ACM, 177–185.

[3] Jiawei Chen, Can Wang, Martin Ester, Qihao Shi, Yan Feng, and Chun Chen. 2018. Social recommendation with missing not at random data. In *Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM'18)*. IEEE, 29–38.

[4] Jiawei Chen, Can Wang, Sheng Zhou, Qihao Shi, Yan Feng, and Chun Chen. 2019. SamWalker: Social recommendation with informative sampling strategy. In *Proceedings of the World Wide Web Conference*. ACM, 228–239.

[5] R. B. Cialdini and N. J. Goldstein. 2004. Social influence: Compliance and conformity. *Annual Review of Psychology* 55, 1 (2004), 591.

[6] Xiangnan He, Xiaoyu Du, Wang Xiang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer product-based neural collaborative filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

[7] Xiangnan He, Zhenkui He, Jingkuan Song, Zhenguang Liu, Yu Gang Jiang, and Tat Seng Chua. 2018. NAIS: Neural attentive item similarity model for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 30, 12 (2018), 2354–2366.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[9] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[10] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining*. IEEE, 263–272.
[11] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the 4th ACM Conference on Recommender Systems*. ACM, 135–142.
[12] Henry Kautz, Bart Selman, and Mehul Shah. 1997. Referral web: Combining social networks and collaborative filtering. *Communications of the ACM* 40, 3 (1997), 63–65.
[13] Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*.
[14] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[15] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational autoencoders for collaborative filtering. In *Proceedings of the 27th International Conference on World Wide Web*. 689–698.
[16] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing* 7, 1 (2003), 76–80.
[17] Guannan Liu, Yanjie Fu, Guoqing Chen, Xiong Hui, and Can Chen. 2017. Modeling buying motives for personalized product bundle recommendation. *ACM Transactions on Knowledge Discovery from Data* 11, 3 (2017), 1–26.
[18] Hao Ma, Irwin King, and Michael R. Lyu. 2009. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 203–210.
[19] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. 2008. SoRec:social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM International Conference on Conference on Information and Knowledge Management*. 931–940.
[20] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*. ACM, 287–296.
[21] Paolo Massa and Paolo Avesani. 2007. Trust-aware recommender systems. In *Proceedings of the 1st ACM Conference on Recommender Systems*. 17–24.
[22] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 452–461.
[23] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550* (2014).
[24] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning*. ACM, 791–798.
[25] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*. 285–295.
[26] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 111–112.
[27] 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, 1 (2014), 1929–1958.
[28] Peijie Sun, Le Wu, and Meng Wang. 2018. Attentive recurrent social recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 185–194.
[29] Jiliang Tang, Huiji Gao, Huan Liu, and Atish Das Sarma. 2012. eTrust: Understanding trust evolution in an online world. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 253–261.
[30] Jiliang Tang, Xia Hu, and Huan Liu. 2013. Social recommendation: A review. *Social Network Analysis and Mining* 3, 4 (2013), 1113–1133.
[31] Jiaxi Tang and Ke Wang. 2018. Ranking distillation: Learning compact ranking models with high performance for recommender system. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2289–2298.
[32] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item silk road: Recommending items from information domains to social users. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 185–194.
[33] Xin Wang, Wei Lu, Martin Ester, Can Wang, and Chun Chen. 2016. Social recommendation with strong and weak ties. In *Proceedings of the 25th ACM International Conference on Conference on Information and Knowledge Management*. ACM, 5–14.
[34] Le Wu, Peijie Sun, Richang Hong, Yong Ge, and Meng Wang. 2018. Collaborative neural social recommendation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* PP, 99 (2018), 1–13.

[35] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*. ACM, 153–162.

[36] Hong Jian Xue, Xin Yu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep matrix factorization models for recommender systems. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 3203–3209.

[37] Xiwang Yang, Yang Guo, Yong Liu, and Harald Steck. 2014. A survey of collaborative filtering based social recommender systems. *Computer Communications* 41, 5 (2014), 1–10.

[38] Tay Yi, Anh Tuan Luu, and Siu Cheung Hui. 2018. Latent relational metric learning via memory-based attention for collaborative ranking. In *Proceedings of the 27th International Conference on World Wide Web*. 729–739.

[39] Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. 2018. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4320–4328.

[40] Tong Zhao, Julian McAuley, and Irwin King. 2014. Leveraging social connections to improve personalized ranking for collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 261–270.

Received March 2019; revised January 2020; accepted March 2020