Neural Process Lett
https://doi.org/10.1007/s11063-018-9831-7

Trust-Aware Collaborative Filtering with a Denoising Autoencoder

Meiqi Wang[1] · Zhiyuan Wu[1] · Xiaoxin Sun[1] · Guozhong Feng[1] · Bangzuo Zhang[1]

Abstract Collaborative filtering is one of the most successful and extensive methods used by recommender systems for predicting the preferences of users. However, traditional collaborative filtering only uses rating information to model the user, the data sparsity problem and the cold start problem will severely reduce the recommendation performance. To overcome these problems, we propose two neural network models to improve recommendations. The first one called TDAE uses a denoising autoencoder to integrate the ratings and the explicit trust relationships between users in the social networks in order to model the preferences of users more accurately. However, the explicit trust information is very sparse, which limits the performance of this model. Therefore, we propose a second method called TDAE++ for extracting the implicit trust relationships between users with similarity measures, where we employ both the explicit and implicit trust information together to improve the quality of recommendations. Finally, we inject the trust information into both the input and the hidden layer in order to fuse these two types of different information to learn more reliable semantic representations of users. Comprehensive experiments based on three popular data sets verify that our proposed models perform better than other state-of-the-art approaches in common recommendation tasks.

Keywords Collaborative filtering · Denoising autoencoder · Social networks · Trust information

1 Introduction

Recommender systems [1] have been applied widely in various types of electronic service systems, such as e-commerce [2] and social networks. These systems can recommend new items that the user might prefer based on the user's history ratings. Two commonly used

✉ Bangzuo Zhang
zhangbz@nenu.edu.cn

[1] School of Information Science and Technology, Northeast Normal University, Changchun 130117, China

Published online: 22 May 2018

methods in recommender systems are content-based filtering and collaborative filtering (CF) [3]. The basic idea of content-based filtering is to recommend similar items to users based on what they liked in the past. By contrast, CF uses the known preferences of a group of users to make recommendations or to predict the unknown preferences of other users. One of the main difficulties that affect CF is the data sparsity problem, i.e., the rating matrices are very sparse. Another difficulty is cold start problem, where recommendations are required for items that no one has yet rated. A possible method to solve above difficulties is to build a hybrid model that fuses CF and content-based filtering where side information is integrated into the training process. The hybrid model aims to supplement the missing ratings by using the side information. For example, Adams et al. [4] proposed the addition of related (side) information regarding venues and dates to predict the scores of professional basketball games. Porteous et al. [5] introduced a Bayesian probabilistic matrix factorization framework that performed regression against side information to predict movie ratings.

Due to the rapid development of social networks, recommendation algorithms that combine trust information have attracted increasing attention recently. Scott et al. [6] noted that user preferences are always influenced by those of their friends, and thus some trust information-based methods have been proposed. Jamali et al. [7] presented a propagation algorithm to model trust relationships. However, only very sparse explicit trust information is available for use in these algorithms. Thus, considering implicit feedback is a good choice to learn user preferences more exactly [8,9]. For example, SVD++ that is based on singular value decomposition (SVD) combined both implicit feedback and the user ratings together [8]. Recently, Guo et al. [9] proposed a new method called TrustSVD that combined both explicit and implicit feedback based on trust relationships as well as ratings to make better recommendations. The success of these methods demonstrates that adding explicit and implicit information can effectively improve the performance of the model.

The proposed methods mentioned above provide many ways of integrating trust information into a recommender system, but these trust-aware models are only extensions of the traditional methods or models. The problem is that these traditional simple linear transformations cannot explore deep semantic connections in a significant manner. Due to the development of deep learning, the effectiveness of neural networks is proved to be better than that of traditional algorithms in tasks such as speech recognition, natural language processing, and image processing. For example, Hong et al. [10] proposed a novel approach for recovering three-dimensional human poses from silhouettes, and this approach improved the traditional methods by employing multiview locality-sensitive sparse coding in the retrieval process. Moreover, they proposed a novel pose recovery method that used nonlinear mapping with a multi-layered deep neural networks, and the recovery error was reduced by 20–25% [11]. As a representative deep learning method, the autoencoder has been used in many applications and attracted increasing attention. Liu et al. [12] proposed a large margin autoencoder to further boost the discriminability by enforcing different class samples to be highly marginally distributed in the hidden feature space and achieved remarkable results. They also proposed a Hessian regularized sparse auto-encoder by incorporating both Hessian regularization and sparsity constraints into the autoencoder, which performed well at preserving the local geometry of the data points and extracting the hidden structure in the data by using sparsity constraints [13]. Furthermore, several studies have used neural networks as powerful tools in recommender systems. For example, Wu et al. [14] proposed a model called collaborative denoising autoencoder, which was a generalization of several well-known CF models. Pan et al. [15] proposed a correlative denoising auto-encoder (DAE) model to learn correlations based on ratings and trust information to provide the top-N recommendation. Deng et al. [16] developed a novel trust-based approach for recommendation in social networks.

In this paper, we propose two trust-aware neural network models based on a DAE to alleviate the data sparsity problem and the cold start problem in CF. The main contributions of the paper include:

- First, we add masking noise to the raw input data to greatly enhance the robustness of the model.
- Second, we combine the rating information and the explicit trust information to depict each user more accurately, where we refer to this model as TDAE.
- Third, we propose to extract the implicit trust relationships between users based on similarity measures in order to supplement the sparse explicit trust information, where the model that integrates both the explicit and implicit trust relationships is called TDAE++.
- Finally, in order to reduce the nonlinear relationship between the trust information and the rating information, we add the trust relationships into both the input layer and the hidden layer of the autoencoder.

The remainder of this paper is organized as follows. In Sect. 2, we introduce some related studies of DAEs, the trust relationships in social networks, and CF with neural networks. In Sect. 3, we describe our proposed models in detail. And the experimental results and analyses are given in Sect. 4. Finally, we summarize the models and suggest potential ways to improve the performance of the models in Sect. 5.

2 Related Work

2.1 Notations

In this paper, we use the following notations:

- i and j denote user i and user j, respectively,
- r_i and r_j are the rating vectors of user i and user j,
- $r_{i,k}$ and $r_{j,k}$ are the ratings of user i and j on item k,
- $\bar{r_i}$ and $\bar{r_j}$ are the mean values of r_i and r_j,
- N and H are the dimensions of the input layer and the hidden layer of the autoencoder.

2.2 Denoising Autoencoder

The proposed model is based on a classical autoencoder [17]. The autoencoder is implemented as a one hidden-layer neural network, which takes a vector $x \in \mathbb{R}^N$ as the input and maps it to a hidden representation via an activation function:

$$y = \rho\left(W^T x + b\right), \tag{1}$$

where W is a $N \times H$ weight matrix and $b \in \mathbb{R}^H$ is a bias vector. The resulting latent representation is then mapped back to a reconstruction vector $\hat{x} \in \mathbb{R}^N$ by:

$$\hat{x} = \rho\left(W' y + b'\right), \tag{2}$$

where $b' \in \mathbb{R}^N$ is a bias vector and $W' = W$ are tied weights for avoiding over-fitting and for enhancing the robustness of the model.

The parameters of this model are trained to minimize the average reconstruction error:

$$\underset{W^T,\, W',\, b,\, b'}{argmin} \frac{1}{m} \sum_{i=1}^{m} l\left(x_i, \hat{x}_i\right), \tag{3}$$

where m is the total number of the users and l is a loss function, such as the squared loss or the cross-entropy loss.

The DAE extends the classical autoencoder by training the data point x from its corrupted version x' to reconstruct the input vector more accurate. The goal of DAE is to force the hidden layer to discover a more robust low-dimensional representation and to prevent it from simply learning the features from the identity function [18]. The corrupted input x' is typically drawn from a conditional probability distribution $p(x'|x)$. And another way of corrupting the input vector is to mask a random fraction of the input by replacing them with zero.

2.3 Trust Relationships in Social Networks

Trust information reflects interpersonal relationships in the real world in a social network, thereby representing the strength of the relationship and the degree of mutual recognition between users. Thus, trust is a powerful tool for optimizing the recommended results and it plays a very important role in recommender systems. Studies have shown that target users would rather believe the recommendations of their trusted friends than those of other online users who share a common hobby [19]. In general, trust relationships can be obtained directly from most social networks based on mutual friends, which are known as explicit trust relationships. However, the explicit trust relationships are not obvious in some e-commerce websites or review websites, and thus it is often necessary to determine the existence of trust relationships between users based on the similarity of their ratings or reviews, where this type of trust relationship is implicit.

In general, similarity measures can be used to assess the relationships between users. Many similarity measures are employed in CF, such as the cosine similarity, adjusted cosine similarity, and Pearsons correlation coefficient. If the ratings of a user can be considered as vectors in the vector-space, then the user preferences can be represented by their rating vectors and the cosine angles between vectors can measure the similarity between users. However, the cosine similarity does not consider the user preferences, so the adjusted cosine similarity solve this problem by subtracting the mean value of the user ratings:

$$\cos'(i,\, j) = \frac{\sum_{k \in L} \left(r_{i,k} - \bar{r_i}\right) \cdot \left(r_{j,k} - \bar{r_j}\right)}{\sqrt{\sum_{k \in I_i} \left(r_{i,k} - \bar{r_i}\right)^2} \cdot \sqrt{\sum_{n \in I_j} \left(r_{j,k} - \bar{r_j}\right)^2}}, \tag{4}$$

where L is the set of the co-ratings rated by user i and user j, and I_i and I_j represent the sets of ratings rated by user i and user j, respectively.

Pearsons correlation coefficient can measure the correlations between users or items, where the range of the results is $[-1, 1]$, and the user or item will be more similar when the value is greater:

$$pcc\,(i,\, j) = \frac{\sum_{k \in L} \left(r_{i,k} - \bar{r_i}\right) \cdot \left(r_{j,k} - \bar{r_j}\right)}{\sqrt{\sum_{k \in L} \left(r_{i,k} - \bar{r_i}\right)^2} \cdot \sqrt{\sum_{k \in L} \left(r_{j,k} - \bar{r_j}\right)^2}}, \tag{5}$$

However, the Pearsons correlation coefficient is less reasonable when the number of co-ratings for users or items is very small. In addition to the three types of similarity described above, Jaccard and the mean squared deviation are two other commonly used methods. However, the Jaccard method only considers the number of co-ratings by the two users without considering the numerical rating given by the users on the items, whereas the mean squared variance is the opposite.

2.4 Collaborative Filtering with Neural Networks

Applications of neural networks have developed rapidly in computer vision, natural language processing, and other fields, but they have received less attention in CF. In a preliminary study, Salakhutdinov et al. [20] addressed the Netflix challenge using restricted Boltzmann machines. In addition, it is proved that the neural networks can discover nonlinear latent factors in heterogeneous data [21], which makes them promising tools for use in CF. Moreover, Strub and Mary [22] and Dziugaite et al. [23] directly trained autoencoders to predict the last ratings. In general, these neural network-based methods obtain better results than the traditional CF techniques. Thus, we consider using the autoencoder to improve the performance of CF.

3 Proposed Model

The detail of our methods is presented in this section. Firstly, we introduce the input data sparsification process employed in integrating the explicit trust information and the implicit trust information. A tuning parameter α is employed to control the influence of adding noise to the training process of the model. Secondly, we present a method to integrate trust information into DAE. Finally, we choose a suitable similarity measure method to extract implicit trust information.

3.1 Input Sparsification

For rating prediction, data sparsity has always been a serious problem that affects the accuracy of the prediction results. Most previous studies that dealt with sparse inputs addressed this problem by precomputing estimates of the missing values [24]. However, we sparsify the inputs using the autoencoder itself [25]. First, the weighted edges between the input layer and the hidden layer can be limited via the random inhibition of the neurons in the input layer in the forward propagation process, thereby making the input vectors sparser. In the same way, some neurons in the output layer can be set to zero to limit the weighted edges between the output layer and the hidden layer in the backpropagation process, too. Finally, the impact of the sparsification operations can be measured by a tuning parameter α in the loss function.

The concrete way of limiting the weighted edges between the input layer and hidden layer is to change the missing values into zero. In order to prevent the autoencoder from always returning zero, we use an empirical loss function that can disregard the loss of unknown values. That is, the missing values are unknown values in the forward propagation process, so the error during the backpropagation process required to train the model based on the reconstruction errors is also ignored. This operation is similar to removing the neurons with missing values in the methods proposed by Salakhutdinovet et al. [26] and Sedhain et al. [27].

Finally, we integrate the method described above as well as the masking noise into the autoencoder. The input sparsification process is illustrated in Fig. 1. The neural network can achieve supervised learning by using this operation. The value of the original data can be overwritten with zero via the addition of masking noise, but the value actually exists and it is assumed to be missing, so the autoencoder is forced to learn the missing value in this setting.

In this case, the input vector x can be considered as including two parts, one is the element x_i' which is obtained by adding noise on x_i and the other is the element x_i which has not

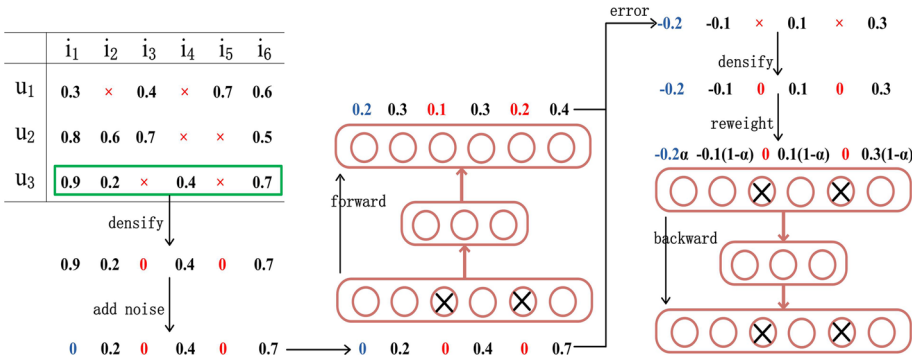

Fig. 1 The process of sparsifying inputs. The input vector is extracted from the user-item rating matrix. First, the missing values of the vector will turn to zero, and then the input is corrupted by masking noise. Before backpropagation, the error of missing values will turn to zero. Moreover, the denoising errors are reweighed by α and reconstruction errors are reweighed by $(1-\alpha)$

changed. The loss function is then modified to emphasize the denoising part of the neural network. The loss function comprises two parts based on a tuning parameter α:

$$L_\alpha\left(x, x'\right) = \alpha\left(\sum_{x'_i \in C(x')}\left[\hat{x'}_i - x'_i\right]^2\right) + (1-\alpha)\left(\sum_{x_i \in N(x)}\left[\hat{x}_i - x_i\right]^2\right), \tag{6}$$

where α is the tuning parameter for the denoising squared error, $(1-\alpha)$ is the tuning parameter for the reconstruction squared error, $x' \in \mathbb{R}^N$ is the corrupted part of x, $C(x')$ is the set of corrupted elements of x, $N(x)$ is the set of the unchanged elements of x, $\hat{x'}_i$ and $\hat{x}_i$ are the ith outputs of the network.

3.2 Integrating Trust Information

In general, CF algorithms perform calculations using the user ratings for a range of items. However, due to the sparsity of the ratings, using the rating information alone is highly restrictive. Therefore, the recommendation performance can be enhanced if more abundant information can be obtained for the user or the item. Thus, we integrate the user trust information in the social networks with the user rating information together in order to improve the accuracy of our proposed method. We employ the DAE described in Sect. 2.2 as our base model in our proposed model. A straightforward approach is to inject the trust information into the input layer to train the model, and then denote the representations in the output layer as the final predicted ratings:

$$\hat{z} = \rho\left(W'\left(\rho\left(W^T\left\{x_i, t_i\right\} + b\right)\right) + b'\right), \tag{7}$$

where $\hat{z}$ is the final representation of the output layer, $W^T \in \mathbb{R}^{H\times(N+T)}$ and $W'^T \in \mathbb{R}^{H\times N}$ are the weight matrices, respectively, $t_i \in \mathbb{R}^T$ is the trust information for u_i , $b \in \mathbb{R}^H$ and $b' \in \mathbb{R}^N$ are the bias vectors, and ρ is a hyperbolic tangent function.

However, Ngiam et al. [28] showed that the correlations between ratings and trust data are highly nonlinear with different distributions. Thus, in order to solve this problem and learn more features from trust information, we inject the trust relationship into both the input layer and hidden layer of the autoencoder:

$$\hat{z} = \rho\left(V'\left\{\rho\left(V^T\left\{x_i, t_i\right\} + b\right), t_i\right\} + b'\right), \tag{8}$$

However, if the dimension of the trust information is excessively large, the autoencoder may have difficulties in utilizing these data effectively. Thus, we enforce a constraint that the dimension of the input layer must be greater than the dimension of the hidden layer, and the latter must be greater than the dimension of the trust information [25], i.e., $N \gg H \gg T$, where N denotes the dimension of the rating information of the input layer, which is the number of the neurons of rating information in the input layer. H denotes the dimension of the rating information of the hidden layer, which is the number of the neurons of the rating information in the hidden layer. And T denotes the dimension of the trust information integrated into the input layer or the hidden layer, which is the number of the neurons of the trust information of the input layer or the hidden layer. Finally, we obtain an autoencoder that can incorporate ratings and trust information, and we train it by backpropagation.

3.3 Extracting Implicit Trust Information

The accuracy of a recommender system can be improved effectively by integrating trust relationships. However, the explicit trust information is generally very sparse. Therefore, implicit trust information has attracted more and more attention. Recently, implicit information has been applied in many methods. For example, Zheng et al. [29,30] proposed a model called IMPLICIT neural autoregressive distribution estimator for collaborative filtering (IMPLICIT CF-NADE) based on the CF-NADE model to exploit the implicit users feedback. And this method performs better than classical implicit matrix factorization [31]. However, exploring more implicit trust information between users is still a challenge. A common way to overcome this challenge is to employ similarity measures to assess the relationships between users. Papagelis et al. [32] proposed a method for measuring the degree of trust between two users using Pearsons correlation coefficient, they considered that this similarity could represent the degree of trust between two people to some extent. However, if there is only one co-rating between two users, then regardless of the rating score, the final similarity is 1, which is not consistent with common sense. Therefore, when calculating the similarity, Wang et al. [33] considered this problem and standardized the similarity to [0, 1]:

$$Sim_{i,j} = \begin{cases} 1, & i = j, \\ \left(1 - \frac{1}{n}\right)\left(\frac{pcc(i,j)+1}{2}\right), & i \neq j. \end{cases} \tag{9}$$

where n is the number of co-ratings between two users. In addition, Wang et al. proposed to use a threshold θ of similarity to infer the implicit trust relationship, the implicit trust relationship can be conformed only when the similarity is greater than the threshold θ:

$$t_{i,j} = \begin{cases} 1, & \text{if } \ Sim_{i,j} \geq \theta, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

where $t_{i,j}$ is the binary implicit trust relationship between user i and user j. In this paper, we use the method proposed by Wang et al. to extract the implicit trust relationship from the user rating information in order to extend the original trust information and obtain a new denser trust matrix. Another point to note is that though the implicit trust relationships are extracted from the rating information, the implicit trust information and the ratings are two types of data. We then integrate the implicit trust relationships into the TDAE model to obtain the TDAE++ model, and the experiments show that the TDAE++ model can produce more

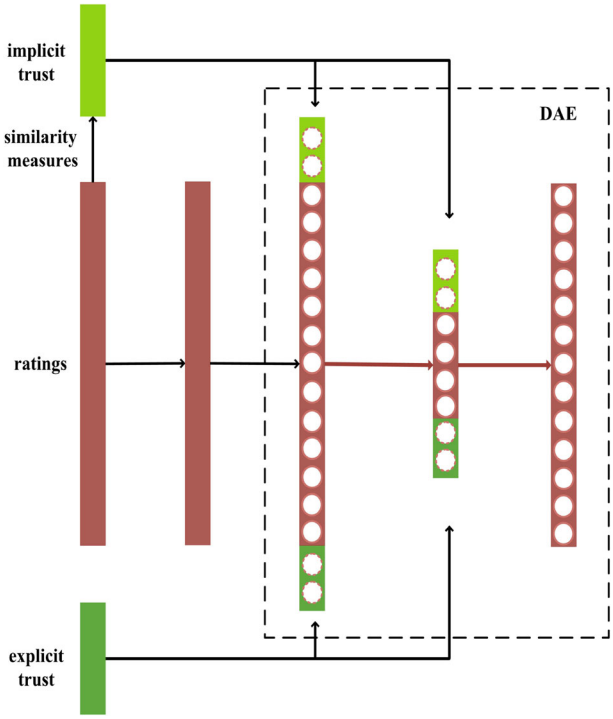

Fig. 2 The structures of TDAE and TDAE++. The autoencoder has two kinds of inputs: one is the rating information and the other is the trust information. The trust information is injected into both the input and the hidden layer of the autoencoder. Moreover, the explicit trust is in the dark green background, and the implicit trust is in the light green background. (Color figure online)

accurate results on three data sets. Moreover, in order to make the structure of the proposed models clearer, we show the overall structure of the models in Fig. 2.

In fact, the most important step in the process of extracting implicit trust information is how to determine the optimal value of θ. In our experiments, θ is closely related to model TDAE++. The process of determining the optimal value of θ is as follows. The implicit trust information is extracted from the rating vectors of two different users, and the larger value of θ, the less implicit trust links can be extracted. We search the different values of θ on TDAE, then observe the experimental results of TDAE++ to get the optimal value of θ.

3.4 Training Loss

We employ the following loss function to train the proposed models, after regularization, the loss function can be written as:

$$L_{\alpha}\left(x, x'\right) = \alpha \left(\sum_{x'_i \in C(x')} \left[\hat{x'}_i - x'_i \right]^2 \right) + (1-\alpha) \left(\sum_{x_i \in N(x)} \left[\hat{x}_i - x_i \right]^2 \right) + \lambda \left(|W|^2_{Fro} \right), \tag{11}$$

where W is the weight matrix and λ is the regularization hyperparameter. λ is used to control the learning degree of the model relative to the sample, which affects the generalizability of the model. When λ has a very small value, the model fits the training set better, but it also increases the risk of overfitting. When λ is zero, this is equivalent to the formula without a regularization term, and the model does not have any protection against overfitting. When λ has a large value, the regularization effect increases, but the risk of underfitting the model also increases. Thus, it is necessary to manually debug the value of λ and finally determine the appropriate value of λ according to the experimental results.

Moreover, during the training process of the model, the weight matrix W is initialized with the fan-in rule $W_{i,j} \sim \left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right]$, and W is then optimized by stochastic gradient descent (SGD), where the mini-batch size is 35. SGD is an effective method for training deep networks and SGD variants can obtain state-of-the-art performance. SGD minimizes the loss by optimizing the network parameter W:

$$W = \underset{W}{argmin} \frac{1}{N} \sum_{i=1}^{N} l(x_i, W), \tag{12}$$

where $x_1, x_2, \ldots, x_N$ are the elements in the training data set. And we will consider a mini-batch $x_1, x_2, \ldots, x_M$ of size M at each step of using SGD. The mini-batch is used to approximate the gradient of the loss function with respect to the parameters by computing:

$$\frac{1}{M} \frac{\partial l(x_i, W)}{\partial W}, \tag{13}$$

The process employed of updating W is as follows:

$$W' = W - \frac{\gamma}{M} \sum_{i=1}^{M} \frac{\partial l\,(x_i, W)}{\partial W}, \tag{14}$$

where γ is the learning rate.

4 Experiments

4.1 Data Sets

We use three popular data sets in our experiments: Filmtrust, Epinions, and Douban. The Filmtrust data set contains 35,497 ratings for 2071 movies provided by 1508 users according to a ratings range from 0 to 4. This data set also includes 1853 trusted relationships provided by 609 users. The Epinions data set contains 40,289 users and 139,738 movies. The total number of movie ratings is 662,824 and there are 487,183 claimed social relationships. The Douban data set contains 129,490 users and 58,541 movies. The total number of movie ratings is 16,830,839 and there are 1,692,952 claimed social relationships. The statistics for these data sets are shown in Table 1.

4.2 Comparisons

We focus on the CF problem, so it is reasonable to compare our proposed models with other methods for rating prediction tasks, such as TrustSVD and DAE. Thus, we select several state-of-the-art algorithms to compare and evaluate our models. Brief descriptions of the baseline models are given as follows:

Table 1 Data set statistics

	Users	Items	Ratings	Trust links	Rating density (%)	Trust density (%)
Filmtrust	1508	2071	35,497	1853	1.1360	0.0814
Epinions	40,289	139,738	662,824	487,183	0.0118	0.0294
Douban	129,490	58,541	16,830,839	1,692,952	0.2220	0.0100

- SVD++ [8]. This algorithm predicts ratings based on the users history browsing data and history ratings as implicit feedback by employing SVD.
- TrustSVD [9]. This is a simple and widely used algorithm for rating prediction. Explicit trust information is added to SVD++ to obtain more accurate predictions.
- Probabilistic matrix factorization (PMF) [34]. PMF employs a probabilistic model based on Regularized MF for further optimization. It can predict unknown values of the ratings matrix based on the user and the item feature matrices.
- TrustMF [35]. The TrustMF algorithm adds trust information based on MF to improve the performance of the model.
- DAE [18]. DAE improves the accuracy of predicting ratings with the classical autoencoder by adding noise to the input data. In addition, this DAE refers to the model for rating prediction in 2016 [25].

4.3 Evaluation Metrics

Two evaluation metrics are used to measure the performance of our proposed models: mean absolute error (MAE) and root mean squared error (RMSE). When the values of the these two evaluation indicators are smaller, the performance of models is better. For a user i, the real rating on item k is $r_{i,k}$, the predicted rating is $\hat{r}_{i,k}$, and T is the number of ratings in the test set.

- MAE is defined as:

$$MAE = \frac{\sum_{i,k} |r_{i,k} - \hat{r}_{i,k}|}{T}, \tag{15}$$

- RMSE is defined as:

$$RMSE = \sqrt{\frac{\sum_{i,k} \left(r_{i,k} - \hat{r}_{i,k}\right)^2}{T}}. \tag{16}$$

4.4 Implementation Details

We consider three data sets to provide the implementation details in our experiments. Two main types of parameters can affect the performance of models: the parameter used to partition the data set and the parameters employed in the training process of the model. Thus, we explain the impact of each parameter in the following paragraphs.

We compare the effects of the proposed models with different training ratios on Filmtrust data set and Table 2 shows that the accuracy of the models improve significantly when the training ratio is 90/10%. Thus, we use 90% of the data set as the training set and 10% as the test set for all of the neural network models.

We train a two-layer autoencoder on each data set to assess the influence of the model parameters, where the neural network is optimized by stochastic backpropagation with a mini-batch of size 35, and a weight decay is added for regularization. The activation functions

Table 2 Influence of the training ratio on Filmtrust data set

	90/10%	% Improv.	80/20%	% Improv.
DAE	0.8156	–	0.8265	–
TDAE	0.7885	3.3227	0.8209	0.6775
TDAE++	0.7656	6.1304	0.8151	1.3793

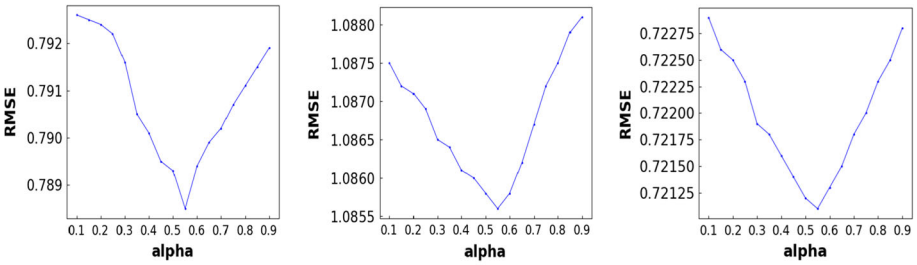

Fig. 3 Influence of alpha (α) on RMSE. **a** Filmtrust. **b** Epinions. **c** Douban

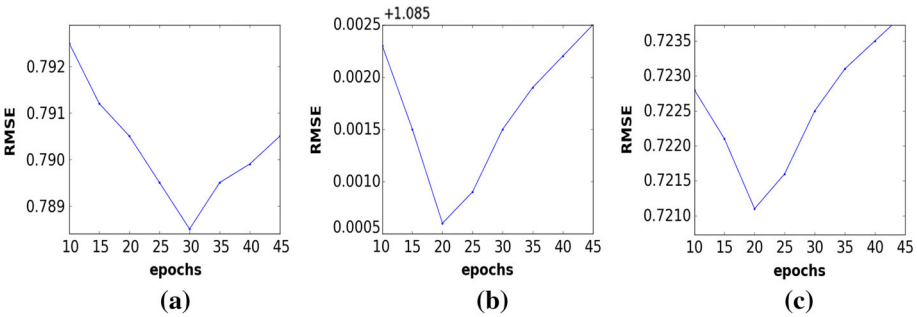

Fig. 4 Influence of epochs on RMSE. **a** Filmtrust. **b** Epinions. **c** Douban

are hyperbolic tangents. We perform a series of comparative experiments on the proposed models to determine the optimal parameters.

In our experiments, we use α as a tuning parameter to constrain the effect of adding noise on the autoencoder. We set the initial value of α to 0.05 and then increase it incrementally to 0.95 at a step size of 0.05. Figure 3 shows the gradual change process of RMSE as the value of α increases. The performance of the neural network is the best when the parameter α is 0.55 on three data sets.

Selecting an appropriate epoch of the model can improve the efficiency of the experiment. Our experiments show that the model works best on Filmtrust data set when the epoch is 30. In addition, for Epinions and Douban data sets, the time cost and the model performance can be balanced to the greatest extent when the epoch is 20. And the process of determining the optimal epoch for each data set is showed in Fig. 4.

The parameter with the greatest impact on the accuracy of the model is the size of the hidden layer, i.e., the number of neurons in the hidden layer, which reflects the ability of the hidden layer to learn the semantic representations of users. The user features can not be extracted if this number is excessively small, however, the transformation between the layers will be similar to the identity transformation if the value is excessively large. In the

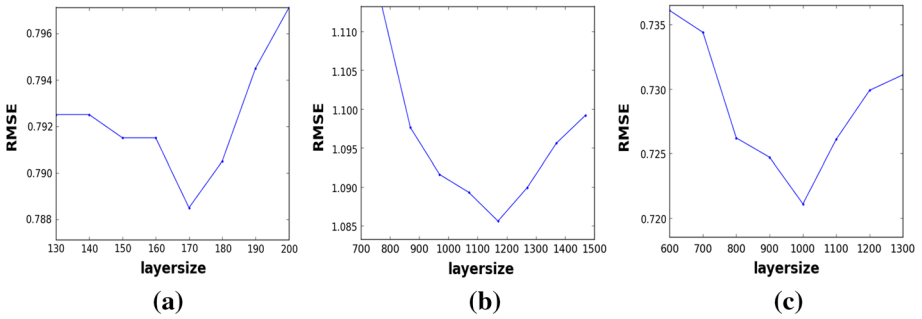

Fig. 5 Influence of layersize on RMSE. **a** Filmtrust. **b** Epinions. **c** Douban

experiments, we employ a two-layer autoencoder in order to determine the optimal number of neurons in the hidden layer. As Fig. 5 shows, the RMSE of the model is the lowest when the layer with 170 neurons on Filmtrust data set. In addition, the best results are obtained when the hidden layer with 1170 neurons on Epinions data set, and 1000 neurons on Douban data set.

4.5 Results and Analyses

We conduct a series of experiments to evaluate the performance of the proposed models. For each data set, the parameter settings of the DAE-based models are the same. All the experiments can be divided into three types. The first type is to compare the effects of the proposed models and the baseline models, the second type is to analyse the influence of the explicit trust information, the last type is to analyse the effect of integrating the implicit trust information. In addtion, we give the results obtained from DAE [25] with the same experimental parameters settings of the DAE-based models on Douban data set.

- Comparison with baseline models

We compare the performance of the proposed models with baseline models in Table 3. As Table 3 shows, DAE performs better than the traditional CF methods. There are two reasons for the results. First, the traditional methods are based on the linear transformation which can not learn the nonlinear relationship from data. However, the model based on neural network can avoid this defect. Since the autoencoder has the ability to find out the deeper and richer semantic information with a nonlinear activation function, it can extract the information from the data more effectively. Second, the neural network can not only exploit the explicit information, but also help to find the implicit trust relationships between users. In addition, it can be seen that the accuracy of TDAE and TDAE++ is better than DAE on Filmtrust data set and Epinions data set. That is, the neural network can model users more accurately after integrating the trust information. However, since the trust density of Douban is too low, it is difficult to use the neural network to learn more accurate user representations by the small amount of explicit trust information. Thus the performance of the model is limited, and the improvement of the proposed models on Douban data set is not obvious.

- Influence of explicit trust information

In Table 4, we compare the effects of TDAE and DAE to illustrate the influence of the explicit trust information. And all experimental parameters of TDAE and DAE are the same on each data set, the only difference between DAE and TDAE is that TDAE integrates the explicit trust information between users. In fact, user modeling by rating information and

Table 3 RMSE with a training/testing set of 90/10%

	Filmtrust		Epinions		Douban	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
SVD++	0.8344	0.6530	1.1214	0.8337	0.7827	0.6144
TrustSVD	0.8312	0.6468	1.1433	0.8532	0.8637	0.6739
PMF	0.9619	0.7099	1.4664	0.9846	0.7644	0.5843
TrustMF	0.9303	0.6971	1.1404	0.9737	0.8391	0.6436
DAE	0.8156	0.6244	1.1001	0.8490	0.7226	0.5704
TDAE	0.7885	0.6071	1.0856	0.8391	0.7211	0.5695
TDAE++	0.7656	0.5905	1.0759	0.8314	0.7199	0.5690

Table 4 Influence of integrating explicit trust information

	Filmtrust		Epinions		Douban	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
DAE	0.8156	0.6244	1.1001	0.8490	0.7226	0.5704
TDAE	0.7885	0.6071	1.0856	0.8391	0.7211	0.5695
% Improv.	3.3227	2.7706	1.3180	1.1660	0.2075	0.1577

by trust information are from different perspectives, thus the integration of these two types of information can obtain more accurate user representations. As Table 4 shows, the effect of adding explicit trust information into TDAE on Filmtrust and Epinions is obvious, which shows that the neural network can obtain more accurate user representations by integrating the explicit trust information. However, the trust density is only 0.01% in Douban data set, thus it is difficult for the neural network to utilize the sparse trust information and the improvement of TDAE on Douban data set is not significant. In addition, the different density of the trust information result in different enhancement of the experimental performance, the improvement in RMSE is higher when the trust density is larger. For example, the RMSE of TDAE is increased by 3.4369 and 1.318% on Filmtrust and Epinions data set respectively, while on Douban data set, its improvement is only 0.2075%. As a result, we can see that our proposed model TDAE performs better on the data sets with denser trust information or richer social information between users.

- Influence of implicit trust information

We extract the implicit trust relationships from the rating information based on a similarity measure. As Table 5 shows, the model TDAE++ performs better than TDAE, which shows that the implicit trust can improve the accuracy of predictions. However, the implicit trust information is only a supplement to the explicit trust information, and it actually plays a role of propulsion, so the performance of model TDAE++ and TDAE on the three data sets is similar. Thus, the effect of TDAE++ improves obviously on Filmtrust data set and Epinions data set, the promotion effect on Douban is not significant. A possible reason is that the user ratings are more focused on the some popular items, and so a part of the extracted implicit trust information is repetitive with the explicit trust information in Douban data set. Therefore, we can see that TDAE++ can get better results on the data sets with large trust information density.

Table 5 Influence of integrating implicit trust information

	Filmtrust		Epinions		Douban	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
TDAE	0.7885	0.6071	1.0856	0.8391	0.7211	0.5695
TDAE++	0.7656	0.5905	1.0759	0.8314	0.7199	0.5690
% Improv.	2.9042	2.7343	0.8935	0.9176	0.1664	0.0877

In addition, the model TDAE++ integrates the trust information on the basis of sparse ratings to enrich each users features, the achieved results can show that the model TDAE++ can also help to alleviate the cold start problem.

5 Conclusion

In this paper, we propose two trust-aware neural network models for CF. In order to alleviate the data sparsity problem and the cold start problem, the proposed model TDAE integrates the user rating information and the explicit trust information together to improve the accuracy of the recommender systems. In addition, in order to overcome the sparsity of the explicit trust information, we use a similarity measure to extract the implicit trust relationships. The model that integrates both the explicit and implicit trust information is called TDAE++.

In future research, some possible enhancements may improve the accuracy of our proposed models. First, other types of information such as comment text and descriptive information for items can be employed as side information. For example, it will be helpful to utilize users comment text to extract the implicit trust information between users in Douban data set. Second, different types of autoencoders such as a stacked autoencoder or sparse autoencoder can be used in the model. Third, other types of neural network can be used, such as convolutional neural network (CNN) or recurrent neural network (RNN).

Acknowledgements This paper is supported by the National Natural Science Foundation of China (71473035, 11501095), Jilin Provincial Science and Technology Department of China (20150204040GX, 20170520051Jh), and Jilin Province Development and Reform Commission Projects (2015Y055, 2015Y054).

References

1. Resnick P, Varian HR (1997) Recommender systems. Commun ACM 40:56–58
2. Schafer JB, Konstan J, Riedl J (2001) E-commerce recommendation applications. IEEE Internet Comput 5:115–153
3. Herlocker JL, Konstan JA, Terveen LG, Triedl JT (2004) Collaborative filtering recommender systems. ACM Trans Inf Syst 4:5–53
4. Adams RP, Dahl GE, Murray I (2010) Incorporating side information in probabilistic matrix factorization with Gaussian processes. Papeles De Poblacin, pp 33–57
5. Porteous I, Asuncion AU, Welling M (2010) Bayesian matrix factorization with side information and Dirichlet process mixtures. M. Fox and D. Poole, AAAI, AAAI Press, New York
6. Scott J (1988) Social network analysis. Sociology 22:109–127
7. Jamali M, Ester M (2010) A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the 4th ACM conference on recommender systems, pp 135–142
8. Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 426–434

9. Guo G, Zhang J, Yorke-Smith N (2015) Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. In: 29th AAAI conference on artificial intelligence. AAAI Press, pp 123–129
10. Hong C, Yu J, Tao D, Wang M (2014) Image-based 3D human pose recovery by multi-view locality sensitive sparse retrieval. IEEE Trans Ind Electron 62(2):3742–3751
11. Hong C, Yu J, Wan J, Tao D, Wang M (2015) Multimodal deep autoencoder for human pose recovery. IEEE Trans Image Process 24:5659–5670
12. Liu W, Ma T, Xie Q, Tao D, Cheng J (2017) LMAE: a large margin auto-encoders for classification. Signal Process 141:137–143
13. Liu W, Ma T, Tao D, You J (2016) HSAE: a Hessian regularized sparse auto-encoders. Neurocomputing 187:59–65
14. Wu Y, DuBois C, Zheng AX, Ester M (2016) Collaborative denoising auto-encoders for top-N recommender systems. In: ACM international conference on web search and data mining. ACM, pp 153–162
15. Pan Y, He F, Yu H (2017) Trust-aware top-N recommender systems with correlative denoising autoencoder. arXiv:1703.01760
16. Deng S, Huang L, Xu G, Wu X, Wu Z (2017) On deep learning for trust-aware recommendations in social networks. IEEE Trans Neural Netw Learn Syst 28(5):1164
17. Scholkopf B, Platt J, Hofmann T (2006) Greedy layer-wise training of deep networks. In: International conference on neural information processing systems. MIT Press, pp 153–160
18. Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th international conference on machine learning. ACM, pp 1096–1103
19. Adomavicius G, Tuzhilin A (2005) Towards the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17:734–749
20. Teytaud O, Gelly S, Mary J (2007) Active learning in regression, with application to stochastic dynamic programming. In: International conference on informatics in control, automation and robotics, ICINCO and CAP, pp 373–386
21. Lecun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444
22. Strub F, Mary J (2015) Collaborative filtering with stacked denoising autoencoders and sparse inputs. In: NIPS workshop on machine learning for e-commerce
23. Dziugaite GK, Roy DM (2015) Neural network matrix factorization. arXiv:1511.06443
24. Base LT (1995) Neural networks for pattern recognition. Oxford University Press, Oxford
25. Strub F, Gaudel R, Mary J (2016) Hybrid recommender system based on autoencoders. In: The workshop on deep learning for recommender systems. ACM, pp 11–16
26. Salakhutdinov R, Mnih A, Hinton G (2007) Restricted Boltzmann machines for collaborative filtering. In: Proceedings of the 24th international conference on machine learning. ACM, pp 791–798
27. Sedhain S, Menon AK, Sanner S, Xie L (2015) Autorec: autoencoders meet collaborative filtering. In: Proceedings of the 24th international conference on world wide web companion, pp 111–112
28. Ngiam J, Khosla A, Kim M, Nam J, Lee H, Ng AY (2011) Multimodal deep learning. In: Proceedings of the 28th international conference on machine learning, pp 689–696
29. Zheng Y, Tang B, Ding W, Zhou H (2016) Neural autoregressive collaborative filtering for implicit feedback. In: Proceedings of the 1st workshop on deep learning for recommender systems
30. Zheng Y, Tang B, Ding W, Zhou H (2016) A neural autoregressive approach to collaborative filtering. In: International conference on machine learning (ICML), pp 764–773
31. Hu Y, Koren Y, Volinsky C (2008) Collaborative filtering for implicit feedback datasets. In: 8th IEEE international conference on data mining. IEEE, pp 263–272
32. Papagelis M, Plexousakis D, Kutsuras T (2005) Alleviating the sparsity problem of collaborative filtering using trust inferences. In: 3rd international conference, iTrust 2005, proceedings DBLP, pp 224–239
33. Wang J, Hu J, Qiao S, Sun W, Zang X, Zhang B (2016) Recommendation with implicit trust relationship based on users similarity. In: International conference on manufacturing science and information engineering (ICMSIE), pp 373–378
34. Mnih A, Salakhutdinov R (2008) Probabilistic matrix factorization. In: Neural information processing systems, pp 1257–1264
35. Yang B, Lei Y, Liu D, Liu J (2013) Social collaborative filtering by trust. In: Proceedings of the 23rd international joint conference on artificial intelligence, pp 2747–2753