School of Computer Science and Engineering

Faculty of Engineering

The University of New South Wales

Dual Trust-Aware Recommender System

by

Guozhao Luo

Thesis submitted as a requirement for the degree of

Bachelor of Computer Science Honours

Submitted: 4 Dec 2019

Supervisor: Dr. Lina Yao

Student ID: z5096182

Acknowledgements

The author would like to thank his supervisor Lina Yao for being a crucial guiding force in the formation, development and shaping of this thesis.

The author would like to thank Manqing Dong for her valuable suggestions.

Abstract

Leveraging the extra information to alleviate the data sparsity and cold start problems in recommender systems has been an active research field in recent years. In several cases, researchers incorporate the trust relationship into consideration. However, most existing trust-based recommender systems only focus on the single level of users' mutual trust relationships, neglecting the quality or accuracy of the data source, which could dramatically degrade the performance of recommendations. To help resolve these issues, we consider the bi-directional dual trust-aware recommender system, by incorporating trustworthiness of source and users' relationships, to provide the reliable, trustworthy resources as a whole to end-users for the better decision-making process. We demonstrate that the proposed model is a generalisation of several well-known deep-learning collaborative filtering models, which utilises a deep autoencoder network to build the recommender model and ultimately improves trustworthiness on both ends. Experimental results on several public datasets show that our approach outperforms some state-of-the-art rating-predicting recommendation methods on a variety of common evaluation metrics.

Abbreviations

AE Auto-Encoder

CDAE Collaborative Denoising Auto-Encoders

DAE Denoising Auto-Encoders

GAN Generative Adversarial Network

MAE Mean Average Error

ReLU Rectified Linear Activation Unit

RMSE Root Mean Squared Error

TARS Trust-based Ant Recommender System

TDAE Trust-aware Collaborative Denoising Auto-Encoders

Contents

List of Figures

Chapter 1

Introduction

With the rapid growth of information available on the World Wide Web, users are often confronted with a serious information overload problem. To alleviate this issue, recommender systems are proposed to provide users with personalised recommendations of information, products or services to satisfy their tastes and preferences [AT05]. Many applications have benefited from recommender systems, including e-commerce sites (Amazon, eBay, Alibaba), streaming services (YouTube, Spotify, Netflix), social media (Facebook, Twitter, Instagram), location-based services (Google Maps, TripAdvisor), and news aggregators (Google News, Apple News).

Traditional recommenders can be generally classified into content-based, collaborative-filtering, and hybrid-filtering methods, among which collaborative-filtering has been the most prominent method [RRS11]. Collaborative-filtering is based on the assumption that people like things similar to other items that they have liked, as well as items that have been liked by other people with similar taste. As the amount of web users drastically increases, two particular challenges facing the recommender systems are the data sparsity problem and the cold start problem. Cold start refers to the lack of recommendations for a new user as well as new items [LKH14], and sparsity occurs when a user produces very few numbers of ratings in a huge dataset so that the data generated is insufficient to identify similarities for recommendations [HCZ04]. Moreover, the classic collaborative-filtering method is often unable to learn the hidden relationships among users and products [ZYST19].

A usual approach to solving the above issues is to consider the extra information sources including social relationships, which is also called the trust relationships, in social networks or context of ratings in the recommendation process. In most cases, such methods are called trust-aware recommender systems. current approaches for trust-aware recommender systems can be generally divided into two directions: memory-based and model-based approaches [JE10]. Memory-based trust-aware recommender systems utilise memory-based collaborative-filtering methods as their basic models. They search the trust networks to obtain trust neighbours for a given user and search the ratings to obtain the similar neighbours, and a recommendation list is produced by combining the predictions of similar neighbours and trust neighbours [MZL+11]. Model-based trust-aware recommender systems adopt model-based collaborative filtering methods as their basic models, among which the matrix factorisation technique is widely used [DHX+17]. Currently, incorporating deep learning in recommender systems is a pervasive trend, as deep learning algorithms in recommender systems can uncover the latent relationship among users, solving several existing challenges.

However, current recommender systems might not produce the reassuring rating predictions as expected [LMY+12], since most recent trust-aware recommender systems ignore the quality of the data resources, yet user-generated ratings can often be unreliable. Due to the webspam phenomenon, the reliability and trustworthiness of user-generated contents are often compromised [Lap12], which poses challenges to the problem of providing high-quality recommendations. For instance, there may have been fake ratings in the dataset that are made by fake reviewers, and they will generate noise for the rating-prediction algorithms. As recommender systems become more and more popular and ubiquitous in modern e-commerce applications, reliable and accurate recommendations are important and necessary to achieve customer's satisfaction, and a deprecated recommendation list may lead to an unpredictable loss.

With such concern, we propose a bi-directional trust-aware recommender system, which can effectively filter the anomaly ratings and provide a rating prediction based on both the user-item relationships and the user-user trust relationships. The proposed model generalises several previously proposed, state-of-the-art deep learning collaborative filtering models (see Chapter 3). We examine and evaluate the various parameter choices by trial and error in our experiments or according to the suggestions in corresponding references, and compare their performance against prior approaches on real-world datasets. Experimental results on several public datasets show that our approach outperforms some state-of-the-art rating-predicting recommendation methods on a variety of common evaluation metrics.

The rest of the paper is organised as follows. Chapter 2 provides the problem definition, explains the background for this thesis, and includes a literature review of prior work on, or related to, the main problems considered in this thesis. Chapter 3 outlines our methodologies for the proposed approaches that we have considered during the construction of the model, based on the problems that we have covered in Chapter 2. In Chapter 4 and Chapter 5, we examine and contrast the results with some state-of-the-art models, and reflect on what components of the model are accurately constructed, and what can be improved.

Chapter 2

Background

In the following section, we discuss the related works in three branches of our model: (1) trust-aware recommender system; (2) recommendation with deep learning; (3) user ratings filtering.

2.1 Trust-aware Recommender System

The recommendation problem in this thesis is to predict the rating that a user will provide to an item that she has not rated based on her historical ratings and her social network. Given a set of users $\mathcal{U} = \{1, ..., n\}$, a set of items $\mathcal{I} = \{1, ..., I\}$, a record of the users' past ratings of items $\mathcal{O} = (u, i, y_{ui})$, and a record of the users' trust relationship network $\mathcal{G}^{\mathcal{S}} = (\mathcal{V}_{\mathcal{P}}^{\mathcal{S}}, \mathcal{E}^{\mathcal{S}})$, where $\mathcal{V}_{\mathcal{P}}^{\mathcal{S}}$ is the set of vertices that represent users, and $\mathcal{E}^{\mathcal{S}}$ is the set of edges, the goal is to predict the rating that a user will give to an unknown item, based on both a user-item rating matrix and a user-user trust information network.

There are various types of social relationships among users (such as friend relationship, colleague relationship, or stranger relationship), through which users interact with each other, and often have similar interests and preferences within a certain social circle. Trust-based recommender system originated to solve the problem of cold-start situations and data sparsity, and has become a dynamic topic in the field of data mining at present. Especially, Ma et al. [MYLK08] first propose a social regularisation method (SoRec) by considering the constraints of social relationships. This model factorises a social graph by U_Z^T, where U is the user feature matrix, and Z is a factor feature matrix with no real implications. They also adopt a function to adjust trust values according to the degrees of involved nodes in the trust network [MYLK08]. As an improved version of SoRec, Ma et al. introduce a social trust ensemble method (RSTE) to linearly combine a basic matrix factorisation model with a trust-based neighbourhood model [MKL09]. In 2011, Ma et al. further propose that the active user's user-specific vector should be close to the average of her trusted neighbours, and use it as a regularisation term to form a new matrix factorisation model (SoReg) [MZL+11].

In terms of randomised approach, Jamali and Ester propose TrustWalker, a random walk-based model which combines trust-based and item-based collaborative filtering approaches for recommendation and outperforms the traditional collaborative filtering approaches in terms of accuracy [JE09]. In 2010, Jamali and Ester promote TrustWalker and propose SocialMF. This model leverages the trust propagation mechanism to model user preference and integrate with matrix factorisation for recommendation. The authors incorporate the mechanism of trust propagation into the probabilistic matrix factorisation model, which improves the prediction accuracy to a large extent [JE10]. Yang et al. propose a circle-based recommendation algorithm CircleCon that extends the utility of SocialMF. The basic idea is that the trustworthiness of a user's friends may vary concerning their specific categories of domains [YSL12]. Furthermore, Yao et al. propose a hybrid method (TrustMF) that combines both a truster model and a trustee model from the perspectives of trusters and trustees, that is, both the users who trust the active user and those who are trusted by the user will influence the user's ratings on unknown items. Based on the observation that a user demonstrates different preferences with roles of truster and trustee, Yao et al. propose the TrustMF algorithm to further improve recommendation performance [YHHZ14].

One of the challenges emerged as to model the multi-faceted friend relationships adequately. Tang et al. consider both global and local trust as the contextual information in their model, where global trust is computed by a separate algorithm [THGL13]. Similarly, Fang et al. stress the importance of multiple aspects of social trust. They decompose trust information into four general factors which are integrated into a matrix factorisation model [FBZ14].

On the contrary, Pan et al. argue that most of the methods above merely enables trust data in their models at superficial levels. They further point out that trust relationships are very complex yet sparse at the same time, thus incapable of supporting deep models like the ones mentioned [PHY17].

More recently, up-and-coming novel methods treat trust data in more complicated and thorough ways to adjust their models accordingly. To handle the data sparsity problem of ratings and trust relationships, the TrustSVD model is proposed by taking both explicit and implicit feedback of user trust and ratings into account for rating prediction [GZYS15b]. Rafailidis and Crestani introduce a recommendation strategy to rank the latent features based on the users' trust and distrust relationships. In particular, two intermediate trust/distrust-preference user's latent spaces is added to capture the correlations of user's preferences with friends' trust and foes' distrust degrees accordingly [RC17]. Lee and Ma present a hybrid approach that combines user ratings as well as social trust and distrust for better recommendations [LM16]. Quite unorthodoxly, Guo et al. develop a multi-view clustering method concentrating on clustering users using both ratings and trust information. They use the k-medoids to acquire clusters and employ a support vector regression model to determine a prediction for a given item, based on user-, item- and prediction-related features. [GZYS15a]

Several works have also incorporated both trust-statements and rating values in an evolutionary process. Bedi and Sharma (2012) propose a method called Trust-based Ant Recommender System (TARS) incorporating a notion of dynamic trust between users on the biological metaphor of ant colonies to select a small set of users most similar to the target user. [BS12]

2.2 Recommending with Deep Learning

In recent years, deep learning technology has become an outstanding research direction in the field of machine learning, which has been widely applied in fields like image processing, speech recognition and online advertising. Meanwhile, recent studies also demonstrate that deep learning algorithms can help uncover the latent relationship among users, and ultimately solve several existing challenges. Applying deep learning techniques into recommender systems has been gaining momentum due to its state-of-the-art performance and high-quality rating predictions. In 1984, Rumelhart et al. [RHW+88] established the idea of Auto-Encoder (AE) for high-dimensional data processing. A classical AE is typically implemented as a one hidden-layer neural network that encodes the input values x using a function f, then decodes the encoded values $f(x)$ using a function g, as shown in Figure 2.1.

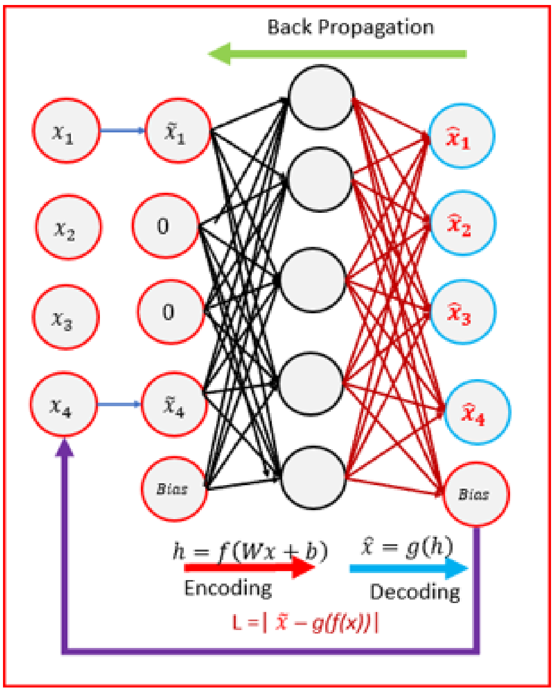

Figure 2.1: The structure of an autoencoder

The objective of AEs is to make the input x and output g as identical as possible, which is represented by the reconstruction error:

$$\arg\min_{W,b} \frac{1}{n} \sum_{i=1}^{n} L(x_i, \hat{x}_i) \tag{2.1}$$

Where L denotes the loss function.

However, if we train the model based solely on the magnitude of the error between the input x and output g, the AE will most likely learn the identity function. To make the AE more useful, researchers created several variances of AE by replacing different constraints. One of them is the Denoising Auto-Encoders (DAE). A DAE tries to reconstruct the original signal from a corrupted input. The DAE is then obtained by adding noise into the input data of the AE, when the AE learns to reconstruct the input data, it is forced to remove the noise to learn the more robust expression of the input data. In this way, the DAE removes corruption of the input and improves the generalisation capability.

AEs for recommender system are mainly used to leverage the compact representations of users or products from sparse rating data for the recommendation. For example, Sedhain et al. [SMSX15] propose AutoRec, which utilises the AE model for recommendation solely based on ratings. We have found that many existing works focusing on utilising neural networks to learn representations of user preferences are often constrained to a single scenario, such as rating prediction [SMSX15], content prediction [WWY15], or tags prediction [WSY15].

The effect that DAE can remove the corruption of input has been applied in different scenarios under different assumptions. For example, the Collaborative Denoising Auto-Encoders (CDAE) model proposed by Wu et al. assumes that user-item interactions observed are a corrupted version of the user's full preference set, and that by integrating DAE into the model, it learns latent representations which help to reconstruct the full user preference [WDZE16]. Pan et al. [PHY17] point out that DAE in recommenders are capable of learning multiple features from different perspectives, and propose a novel Trust-aware Collaborative Denoising Auto-Encoders (TDAE) model which employs DAE to learn hidden features by reconstructing the rating vector and trust vector, and the importance of the two vectors is balanced through a weighted hidden layer in the middle. Rating prediction is made based on the learned hidden vector [PHY17].

Restricted Boltzmann machine (RBM) is the earliest neural network model applied to recommender systems, and several notable trust-based recommender systems adopted RBM for predicting item ratings. The Boltzmann machine (BM) is a stochastic generative neural network that normally has two layers, an input layer which is also known as the visible layer, and a hidden layer. BM can study the complex patterns in the data and performs a strong ability of unsupervised learning. However, the training process is often very time-consuming. To accelerate this process, Sejnowski [Smo86] proposed the restricted version of BM, where the neurons in each layer communicate only with the neurons in the other layer, and not with neurons in the same layer, hence no intralayer communications are made among the neurons, as shown in figure 2.2. A recommender system can learn the probability distribution of ratings, given their previous ratings and the ratings of users to which they were most similar to, and RBM recommender system can use the learned probability distribution to predict ratings for never-before-seen items.

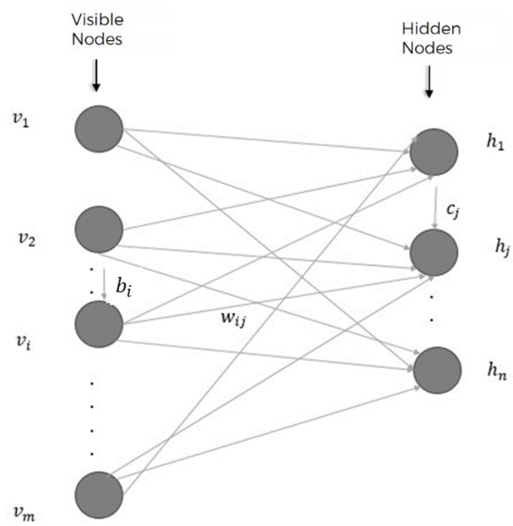

Figure 2.2: The structure of an RBM

An exemplary trust-based recommender system that utilises RBM is introduced by

Nguyen and Lauw [NL16], where the influence of trust relationship on recommendation results is modelled by two methods, one is to take the trust relationship and user score as the observed variables of RBM, and model the trust-influenced by fitting trust relationship and user score at the same time; the other is to add a hidden layer in RBM, on which the trust influence modelling can be realised through weight sharing among users.

After comparing the two above approaches, we conclude that the AE structure may perform better learning user preferences based on rating and trust data. Compared with RBM approaches, whose trust-based recommendation strategies mostly focus on treating the rating and trust information as a whole, we focus on the combination of trust and rating which can allow adjustable importance of trust information during rating prediction.

2.3 User Ratings Filtering

The webspam phenomenon has accompanied the Internet from the very beginning, malicious and inappropriate contents can pervade any information system, and our rating-based recommender system is prone to be affected [KKK17]. Major search engine companies like Google has placed the combat with spam as its priority [Cut11]. One of the worse situations is that on the Internet, company-employed "bots" can publish false information on multiple applications to gain influence for personal interests. For example, the "bots" may be hired to publish favourable comments for a mediocre movie, thus misleading recommender systems to recommend the film to more real users, and increasing its exposure on the platform more than it should. These problems are increasingly rampant in many applications, which have caused a significant challenge to the recommender systems. If unregulated, spam information can deteriorate the quality of recommendation results and deprive legitimate products of exposure that they might earn in the absence of spam. Moreover, spam weakens the credibility of the application and may lead to a lower amount of ratings made by real users, or even less interaction with the application overall.

Numerous detection methods have been deployed to identify these spammers from the system. By reviewing users' behaviours, content similarity and rating patterns, these methods are designed to identify the spammers from the online systems [LMY+12].

A classic and simple way of spam detection is content-based anti-spam algorithms [SH12]. A simplified example would be in email frameworks, clients can keep a white list and blacklist, and distinguish spammers by recognising the characteristics of the mail sender. It can also be accomplished by first analyse the content of the email, then filter the spammers through the keywords in the email. However, with the complexity of user-created contents growing, such naïve anti-spam algorithms can be easily bypassed by spammers, or even manipulated to be used in their interests.

As a highly capable and adaptive tool, deep learning has been integrated in several latest spam detection algorithms. Liu et al. proposed a novel outlier detection algorithm based on unsupervised adversarial network: SOGAAL, which can directly generate informative potential outliers to solve the lack of information caused by sparsity of data in high-dimensional space. This algorithm can advance itself by generating outliers (spam) based on the mini-max game between a generator and a discriminator. [LLZ+19]

Despite that there has been a colossal measure of research on spammers in complex systems, there is as yet an absence of methodical approach to mitigate the impacts of spammers on trust-based recommender systems. Most existing trust-based recommender systems only focus on the single level of users' mutual trust relationships, dismissing the possibility that real-world spammers in the system are compromising the quality and accuracy of the data source.

Chapter 3

Method

Towards the problems discussed in previous chapters, we introduce a variant of dual trust-aware model based on the TDAE model by Pan et al. [PHY17] We presume that the original user's item preference input is randomly corrupted, and by training the trust-aware model, as we will see, the TDAE model is competently bi-directional trust-aware model, as it not only incorporates users' trust relationship information, but also increases the trustworthiness of users' ratings. Furthermore, we will also analyse how such model can drastically mitigate the cold-start and data sparsity issues that have been beleaguering other collaborative-filtering recommender systems for years.

3.1 Problem Definition

In general, recommender models are defined as

$$\hat{y}_{ui} = \mathcal{F}_\theta(u, i) \tag{3.1}$$

where $\hat{y}_{ui}$ is the predicted rating of user u on item i, and θ denotes the model parameters we need to learn from training data.

Suppose that a trust-aware recommender system includes m users and n items. Let R denote the user-item rating matrix, where each entry $r_{u,i}$ represents the rating given by user u on item i, and let I_u denote the set of items rated by user u. On the other hand, suppose that a social network is represented by a graph $G = (V, E)$, where V includes a set of m nodes (users) and E represents the directed trust relationships among users. We can use T to describe the structure of edges E, where $t_{u,v}$ indicates the extent to which user u trusts v. Our goal is first filtering the anomaly ratings $r^*_{u,i}$, and then predict the missing rating values based on the filtered rating matrix R and trust relationship T. Normally, the ratings are integers and fall into [0,5].

The proposed dual trust-aware method can be divided into two main components: filtering the anomaly ratings, and predicting the ratings. The following sections showcase how our model is developed around these two points accordingly, and how its bi-directional trust-aware capability is realised.

3.2 Achieving Dual Trust-aware

3.2.1 Filtering Anomaly Ratings

To achieve the dual-trust system as desired, we initially proposed the unsupervised generative outlier (GAN) detection algorithm to combat the webspam phenomenon. However, after extensive research, we have concluded that adopting GAN model for spam detection is a very novel field and little literature support can be found. It would be time-consuming to develop such a recommender system and even more difficult to retrofit the model for every new dataset. Such a recommender system is complex since it adds unnecessary maintenance overhead and generates retraining costs. Since assuring the quality and accuracy of the data source is a crucial step, it still poses challenges to the problem of providing high-quality recommendations.

To avoid adding unnecessary complexity by requiring two steps to do a “simple” thing, we turned our insight into the traits of DAE models. The main thought of DAE is void repeating extraction of the same feature. DAE can improve the robustness of the extracted intermediate representation to feature noise of the input. It is reasonable to presume that the original user's item preference input is randomly corrupted, as the CDAE model by Wu et al. makes a similar presumption that preferences observed in the rating matrix are a corrupted version of the user's full preference set [WDZE16].

We confirm that the DAE model alone can reconstruct the original model when the corruption is not severe. Therefore, the TDAE model is bi-directional trust-aware, as such model not only incorporates users' trust relationship information, but also increases the trustworthiness of users' ratings. Moreover, DAE also handles the data sparsity problem, since the basic idea of any AE is to reconstruct data through a narrow bottleneck, where any sparsity issue in the original input is elegantly resolved.

3.2.2 Predicting Ratings

To maximise the use of the DAE model, we fuse trust information into the recommender system using the Trust-aware Collaborative Denoising Auto-Encoders (TDAE) model. This model uses DAE to learn ratings' and trusts' hidden vector by reconstructing the users' rating vector and trust relationship vector and balances the importance of the two types of hidden vectors through a weighted hidden layer in the middle, as shown in Figure 3.1.

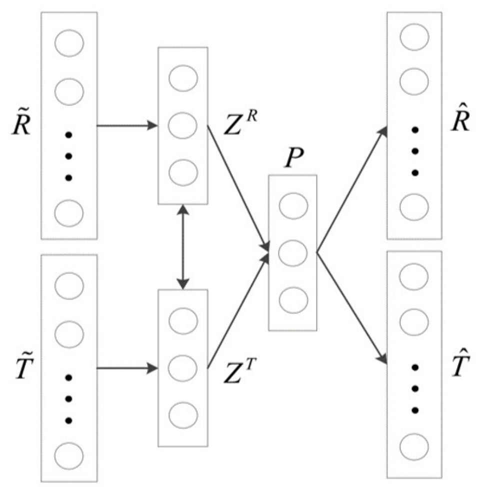

Figure 3.1: The structure of TDAE

Rating predictions are made based on the learned hidden vector. In addition, an association regularisation term is used to avoid over-fitting problem caused by sparse data [PHY17].

To predict the filtered ratings and trust information, we fuse such two hidden vectors via an encoder-decoder structure, and use the latent representation for the final prediction.

By combining trust information and ratings, we have effectively alleviated the cold-start problem, where users may have only rated a few items, or rarely no items at all. Since the reconstruction process of an AE can help to learn more reliable user-specific latent feature vectors, the TDAE model ensures that the user-specific vector can be trained and learned with the ratings matrix as well as the trust matrix.

3.3 Implementation

3.3.1 Denoising Auto-Encoders

To further evaluate the denoising feature of the DAE model, We have recreated the DAE method by randomly corrupting the ratings, based on the TensorFlow library against the FilmTrust dataset, which includes 35,497 ratings from 1,508 users on 2,071 movies, ratings are integer values ranging from 1 to 5, with a sparsity of 98.86% [GZYS13].

In practice, we choose the Adam optimiser to minimise the loss function, which is a stochastic optimisation method. It combines the heuristics of both Momentum and Root Mean Square Propagation. The Adam optimiser update equations for each trainable parameter w is as follows, where η denotes the initial learning rate, g_t denotes the gradient at time t along w, v_t denotes the exponential average of gradients along w, s_t denotes the exponential average of squares of gradients along w, and β_1, β_s are hyperparameters:

$$v_t = \beta_1 * v_{t-1} - (1 - \beta_1) * g_t \tag{3.2}$$

$$s_t = \beta_2 * s_{t-1} - (1 - \beta_s) * g_t^2 \tag{3.3}$$

$$\Delta w_t = -\eta \frac{v_t}{\sqrt{s_t + \epsilon}} * g_t \tag{3.4}$$

$$w_{t+1} = w_t + \Delta w_t \tag{3.5}$$

The idea of "tied weights" is also adopted here, which simply means parameter sharing. Since the encoding and decoding layers of the DAE mirror each other in structure, the shared parameters between each structure allow DAE to learn just one set of weights. The decode weights are the transpose of the encode weights. This method is preferred over learning separate weights for both phases because it reduces the number of parameters and the model can be trained faster. "Tied weights" is also seen as a form of regularisation that leads to better performance in practice.

Furthermore, we also employ the L2-norm loss function in our model as a regularisation term in order to prevent the coefficients from overfitting. It basically minimises the sum of the square of the differences between the target value and the estimated values. Algorithm 1 shows how DAE uses tied weights and L2-norm loss function to do the training and optimisation procedure.

After fine-tuning the DAE model with features such as L2-norm loss function and tied weights, we train our model under the FilmTrust dataset, with ratios of corruption {0.1, 0.3, 0.5, 0.7, 0.9}, which denotes how much of the ratings has been set to a random integer ranging from [0, 5]. Note that at this stage, only the rating values are used. We choose such dataset for its small capacity and comparably low data sparsity, as testing the denoising capability of DAE is our main purpose.

Algorithm 1 Training Denoising Autoencoder

procedure DAE TRAINING($e, b, \mathbf{x}, c, l, \theta$)

$\mathbf{x} = [x_1, x_2, ..., x_n] \in R^{n*m}$ is the input matrix, in which $\mathbf{x_i} \in [0, 5]^m$ is a single input data

e is the amount of epochs to be iterated

b is the amount of batches

l is the learning rate

c is the corruption level

λ is the regularisation rate of L2-norm function

$\theta = \{W, \mathbf{b}, \mathbf{b_h}\}$ where $W \in R^{n*d}, b \in R^d, \theta$ is the parameters of a DAE

for 0 to e **do**

for 0 to b **do**

$\tilde{\mathbf{x}}$ = getCorrputedInput$\{\mathbf{x}, c\}$, where c is the corrupted level

$\mathbf{h} = relu(\tilde{\mathbf{x}} * W + \mathbf{b})$

$\widehat{\mathbf{x}} = relu(\mathbf{h} * W^T + \mathbf{b_h})$

$L(\mathbf{x}, \widehat{\mathbf{x}} = -\sum_{i=1}^{d}[\mathbf{x_i} log \widehat{\mathbf{x}}_\mathbf{i} + (1 - \mathbf{x_i}) log(1 - \widehat{\mathbf{x}}_\mathbf{i})]$

$cost = mean(L(\mathbf{x}, \widehat{\mathbf{x}})) + \lambda \sum_{i=1}^{k} w_i^2$

g = compute the gradients of the cost with respect to θ

for θ_i, g_i in (θ, g) **do**

$\theta_i = \theta_i - l * g_i$

end for

end for

end for

end procedure

To evaluate the rating-denoising accuracy, we adopt the Root Mean Squared Error (RMSE) metric and Mean Average Error (MAE) metric:

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in testset}(r_{ui} - \hat{r}_{ui})^2}{|testset|}} \tag{3.6}$$

$$MAE = \frac{\sum_{(u,i)\in testset}|r_{ui} - \hat{r}_{ui}|}{|testset|} \tag{3.7}$$

Experiments are run 10 times and the average results are reported. The respective RMSE and MAE results are shown in figure 3.2

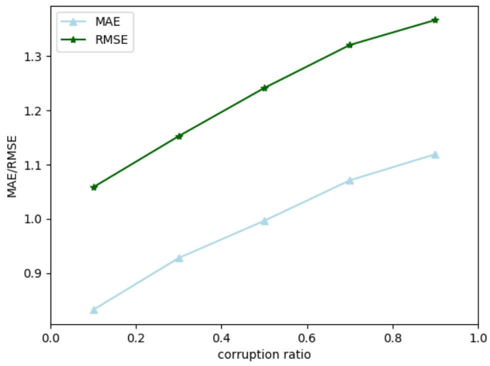

Figure 3.2: DAE model performance on FilmTrust dataset with different corruption ratios

We conclude that using the DAE model alone can reconstruct the original rating model adequately when the corruption level is not severe. Therefore, it is evident that the TDAE model can be considered as dual-trust aware, as such model not only incorporates users' trust relationship information, but also increases the trustworthiness of users' ratings.

3.3.2 The TDAE Model

In the original TDAE model by Pan et al. [PHY17], DAE is employed along with dropout noise. However, as we have just discussed, it is reasonable not to employ any noise of any form in the model, since the real-world ratings input is most likely noisy from spammers or unreliable users. Moreover, drop-out noise is most commonly used for top-N recommendation, not rating prediction.

Given a user-item rating matrix R and a user-user trust matrix T, the corresponding ratings and trust inputs are converted into low-dimensional space with an encoder layer, which is given by

$$Z_u^R = f(W^T \hat{R}_u + b) \tag{3.8}$$

$$Z_u^T = f(V^T T_u + c) \tag{3.9}$$

where $\hat{R}_u$ denotes the corrupted version of rating matrix for u, T_u denotes the trust matrix data for user u, Z_u^R and Z_u^T represent the latent user preferences of u, which is learned from rating and trust data respectively. Note that we assume that the rating matrix has been corrupted from the beginning. Parameters $W \in \mathcal{R}^{m \times k}$, $V \in \mathcal{R}^{n \times k}$ are being trained to learn their corresponding user preferences.

Then, a weighted layer is proposed to integrate these two kinds of representations. A straightforward approach is to directly concatenate representations from rating and trust data for each user. However, a critical issue arises that both the ratings and trust relationships are very sparse and can face severe overfitting problem, yet there is no simple linear relationship between them. In other words, either the trust vector or the rating vector can have a much higher variance, which will make it the dominant influence on the outputs, even if the other one may contain important information. Hence, the authors of the TDAE model propose a correlative regularisation term to build the relations between these two types of information. They build the relation between rating and trust data as simply adding a weight parameter (to formulate the relationship as a linear relationship), and add the error for this relationship as a regularisation term:

$$P_u = \alpha Z_u^R + (1 - \alpha) Z_u^T \tag{3.10}$$

Where α is an adjustable term ranging from [0, 1] to leverage the importance of these two types of representations. The final objective function is:

$$L_T = l(R, \hat{R}) + l(T, \hat{T}) + \frac{\lambda_T}{2} \Omega(W, W', V, V', b, b', c, c') \tag{3.11}$$

where Ω is a regularisation term, and $l(*,*)$ is a loss function.

To estimate the skill of our variant of the TDAE model, we use k-fold cross-validation. Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. This method has a single variable k, which refers to the number of groups to be divided into a given data sample. By selecting a specific value for k, it can be used in the template comparison instead of k, like $k = 10$ is a 10-fold cross-validation. In applied machine learning, cross-validation is predominantly used to approximate the performance of a model of machine learning on unseen data. That is, by using a limited dataset, we can estimate how the model is predicted to perform when used to predict data not used during the model's training. It is a widely used because it usually results in a less biased and less positive approximation of the ability of the model than other approaches, such as a simple split between train and test.

The general procedure is shown in Algorithm 2.

Algorithm 2 K-fold Cross-Validation

```
procedure Cross-Validation
    Shuffle the dataset randomly
    Split the dataset into k groups
    for group i in k do
        Take i as a hold out or test set
        Take k − 1 remaining groups as a training set
        Fit a model on the training set and evaluate it on the test set i
        Summarise the performance of the model using the sample of model evaluation
scores
    end for
end procedure
```

Importantly, for the purpose of the procedure, each observation in the data sample is allocated to a particular group and remains in that group.

Chapter 4

Experimental Results

In this chapter, we report our preliminary results in great details, then compare them with several state-of-the-art recommender system models, and shed light on further improvements and variations for our model.

4.1 Details on Hardware, Software and Dataset Used

We use the group GPU accelerated computing server "DrStrange" for neural network architectures design, training and testing. "DrStrange" is a system/minicomputer customised for the purpose of conducting massive computing tasks, specifically floating point computation intensive deep learning tasks, which makes it very suitable for data-intensive deep neural network training and testing tasks. It currently consists of 2 12-core/24-thread Intel Xeon CPU E5-2697 v2 CPUs, 6 NVIDIA TITAN X Pascal GPUs, a total 768 GiB memory, and an Intel P3608 SSD (1.6TB, NVMe PCIe 3.0 X8 HET MLC) with 7 Seagate Enterprise Capacity 2.5 HDDs (2TB SATA 6Gb/s 7200rpm 128MB).

Tensorflow is a digital computing software open library that uses data flow graphs. The nodes in the graph represent the math operation that reflects the edge between nodes in the graph representing the multi-dimensional tensor. TensorFlow can run on multiple

CPUs and GPUs, and is commonly used for neural network building and learning. For this thesis, we adopt Tensorflow to build our neural network.

Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. We use Pandas to preprocess the data for our neural network.

FilmTrust is a small dataset crawled from the entire FilmTrust website in June, 2011 by Guo, Zhang, and Yorke-Smith [GZYS13]. CiaoDVD is a dataset crawled from the entire category of DVDs from the dvd.ciao.co.uk website in December, 2013 by Guo et al.[GZTYS14] Details on these two datasets are shown in Table 4.1. Our training and testing process are mostly performed on subsets or the entirety of these two datasets.

Feature	**FilmTrust**	**Ciao**
users	1,508	7,375
items	2,071	99,746
ratings	35,497	280,391
density	1.14%	0.03%
trusters	609	6,792
trustees	732	7,241
trusts	1,853	111,781
density	0.42%	0.23%

Table 4.1: Statistics of the two datasets

After further investigation on the two datasets' general statistics, we notice that the density of trust is much smaller than that of ratings in FilmTrust, whereas trust is denser than ratings in Ciao. In this regard, our trust-aware recommender system cannot focus too much on either trust or rating. Both ratings and trust are very sparse across all the data sets, adequately represents the real-world epidemic of data sparsity problem.

4.2 Parameter Tuning

We employ a multilayer DAE with 5 hidden layers, with the middle layer consisting of the lowest dimension, and a descending and ascending number of nodes for the encoding and decoding layers respectively. We use ReLU as the activation function.

Parameter 1: Number of Hidden Layers and Number of Neural Nodes

A basic Autoencoder has three layers: an input layer, a hidden (encoding) layer, and a decoding layer.

As the number of users and items in the datasets is gigantic, simply using 1-3 hidden layers cannot effectively mitigate the data sparsity problem, and the TDAE model cannot learn the useful latent features. On the other hand, we can make the TDAE very deep by increasing the number of layers and nodes per layer, but for very deep layers, the autoencoder will simply learn to copy its inputs to the output, without learning any meaningful representation. It will just mimic the identity function. Training and testing deep AEs can also get burdensome.

To test the most productive and manageable sizes for the TDAE model, we have tested the TDAE model on the FilmTrust dataset, for a range of hidden layers and numbers of neural nodes for each layer, using the 5-fold cross validation method, and report their best RMSE and MAE accuracy. In every test, we have set the weight parameter balancing the importance between trust and rating vectors $\alpha = 0.5$. To inhibit the model from learning the identity function, the number of neural nodes decrease exponentially. The results are shown in Table 4.2.

Hidden layers' dimensions	RMSE	MAE
256, 128, 256	0.981	0.762
512, 256, 128	1.082	0.815
256, 128, 64, 128, 256	0.785	0.626
512, 256, 128, 256, 512	0.788	0.629
512, 256, 128, 64, 128 256, 512	0.790	0.628

Table 4.2: Performance Comparison of Hidden Layers with Different Dimension Choices

We conclude that using three hidden layers results in the worst overall performance, because the model only learns insufficient latent features, whereas for over five number of hidden layers, there is little evidence that extra meaningful features are fully learned. Additionally, large dimension of the first hidden layer in the early stages of neural network, which is directly linked to the dimension of items in the dataset, may slightly hamper the network from essentially being mapped to a lower dimension.

Parameter 2: Activation Function

Since we employ a deep neural network in the TDAE model, to train such model, we apply stochastic gradient descent algorithm with backpropagation of errors to learn the complex trust relationships in the data.

For neural networks in general, nonlinear activation functions are preferred as they allow the nodes to learn more complex structures in the data and receive useful gradient information. Two widely used nonlinear activation functions are the sigmoid and hyperbolic tangent activation functions (tanh). However, two issues plaguing these functions are preventing deep (multi-layered) networks from learning effectively. One of them is the vanishing gradient problem. As error is backpropagated through the network, it decreases dramatically with each additional layer, stopping neurons in deep layers from updating their weights. Another problem is saturation. As these functions are sensitive to changes around their mid-point of their input, such as 0.5 for sigmoid and 0.0 for tanh, large inputs, however, are capped at 1.0 and small values capped at -1 or 0 for tanh and sigmoid respectively.

The rectified linear activation unit (ReLU) resolves these issues by acting as a linear function but functioning as a nonlinear function. It is also sensible to change to the activation sum input, avoiding easy saturation. The rectifier function is also computationally simple since it is a simple calculation that returns the value provided directly, or the value 0.0 if the input is 0.0 or less.

Because of the benefits ReLU possesses, it is the default activation function during training and testing process of our TDAE variant.

Parameter 3: Weight Parameter

As we have shown before, a weighted layer is proposed to integrate the ratings representation and the trust representation, and an adjustable term α is used to leverage the importance of these two types of representations.

$$P_u = \alpha Z_u^R + (1 - \alpha) Z_u^T \tag{4.1}$$

To select the most balanced value for α, we train our model under varying weight parameters on the FilmTrust dataset, with the best resulting hidden layer choice {256, 128, 64, 128, 256}, using the 5-fold cross validation method. The respective RMSE and MAE results are shown in Figure 4.1

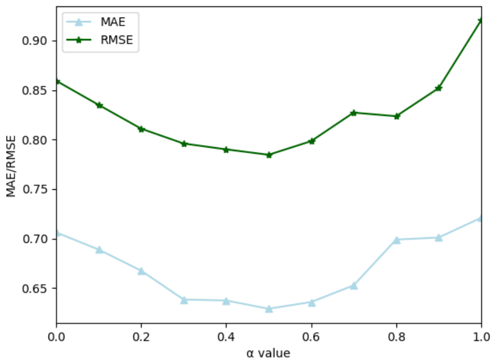

Figure 4.1: TDAE model performance on FilmTrust dataset with varying weight parameters

It is evident that for the FilmTrust dataset, setting the α value to around 0.5 results in the best performance, meaning that the trust and rating representations should be given around the same importance when fusing these two features together.

4.3 Results Comparison

Since we implemented the variant of TDAE model focusing on the rating prediction problem, it is reasonable to compare the experimental results with other models for rating prediction task, such as SoRec, SoReg, SocialMF, TrustMF and TrustSVD.

The following models have been evaluated on the FilmTrust dataset, data obtained by Guo, Zhang, and Yorke-Smith [GZYS15b]. The following trust-based models are compared: SoRec [MYLK08], SoReg [MZL+11], SocialMF [JE10], TrustMF [YLLL16], TrustSVD [GZYS15b]. The comparison models' reporting outcomes are collected by Guo, Zhang, and Yorke-Smith [GZYS15b], all models are tested under the same datasets, using the same 5-fold cross-validation method. In our experiments, we tune the hyperparameters such as batch size and α value by trial and error or according to the suggestions in corresponding references, and report the best results for comparison. The experimental results are presented in Table 4.3.

All	Metrics	SoRec	SoReg	SocialMF	TrustMF	TrustSVD	**TDAE**	Improve
FilmTrust	RMSE	0.810	0.878	0.837	0.810	0.789*	**0.785**	0.51%
	MAE	0.628	0.628	0.674	0.638	0.611*	**0.626**	-2.12%
Ciao	RMSE	1.013	1.183	0.981	0.983	0.955*	**0.941**	1.49%
	MAE	0.765	0.899	0.749	0.742	0.723*	**0.739**	-2.21%

Table 4.3: Performance comparison, where * indicates the best performance among all the other methods, and column 'Improve' indicates the relative improvements that our variant of TDAE achieves relative to the * results.

It is clear that our implementation of the TDAE model performs better than several state-of-the-art rating-predicting models, and is around the same as the best scoring method, namely TrustSVD. However, such a result is inconsistent with Pan, He and Yu's experimental outcome, where the authors claim a 7% improvement [PHY17]. This implies that the trust-based DAE approach may only have a slight lead comparing to other well-performing ratings-only approaches. It has been pointed out that even small improvements in MAE and RMSE may lead to significant differences of recommendations in practice.

4.4 Ablation Studies

We have compared our variant of TDAE model with the DAE model, and the TDAE model that contains no trust information, as shown in section 4.2 and section 3.3.1. Experimental results demonstrate that the TDAE model is around 30 to 50 percent more accurate in terms of RMSE and MAE metrics. We have discussed why employing trust information in our model can improve the performance of recommender systems in Chapter 2.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This thesis focuses on building a dual trust-aware recommender system to alleviate the cold-start and data sparsity problem in classic recommender systems, and to improve the trustworthiness of the ratings.

Our main contributions include:

1. We have conducted extensive analysis of previous trust-aware recommender systems, selected one of the most novel algorithms, namely the TDAE model, constructed a new variant of the model, and validated the algorithms' dual trust-aware features in various ways. It took a significant amount of time to rebuild a model as Pan, He and Yu's paper has showcased [PHY17]. Our implementation on the variant of TDAE is mostly consistent with the original model, as the experimental results demonstrate that the proposed model outperforms several ratings-only recommender systems.

2. We have researched into employing deep learning for filtering anomaly ratings, and after calculating on the time and space complexity it would take to build a completely novel GAN-based fake review detection model, we have chosen to

utilise a simpler approach and conducted several experiments to validate the performance of the DAE model. We ultimately conclude that the DAE model can improve the trustworthiness of the rating input and that the TDAE model itself can be bi-directional trust-aware, which is not mentioned in the original paper.

5.2 Future Work

With respect to the concepts and ideas that we introduce in this thesis, we have done and are willing to do many experiments involving them. However, due to limited time and resources we have, some ideas did not come to fruition. Therefore, we would like to point out some issues in our thesis for future works.

1. We have done some basic experiments on the GAN model for filtering anomaly ratings in this thesis. However, due to the complexity of the model and limited time, we could not finish the model. We are looking forward to researching on how the performance of GAN differs from other spam-detection algorithms in the future.

2. The TDAE model only takes the rating vector into account while calculating the loss function. It would be very worthwhile to enable the trust vector when calculating the loss function.

3. Iterative Vote Aggregation Algorithm [AI14] can be used to calculate the trustworthiness of a user, based on how many ratings such user has made, and how much the user's ratings are in alignment with the prevailing sentiment of the community. In this way, trust information can be constructed even on datasets without given trust information.

Bibliography

[AI14] Mohammad Allahbakhsh and Aleksandar Ignjatovic. An iterative method for calculating robust rating scores. *IEEE Transactions on Parallel and Distributed Systems*, 26(2):340–350, 2014.

[AT05] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, (6):734–749, 2005.

[BS12] Punam Bedi and Ravish Sharma. Trust based recommender system using ant colony for trust computation. *Expert Systems with Applications*, 39(1):1183–1190, 2012.

[Cut11] Matt Cutts. Google search and search engine spam, 2011. https://googleblog.blogspot.com/2011/01/google-search-and-search-engine-spam.html/, Last accessed on 2019-11-28.

[DHX+17] Shuiguang Deng, Longtao Huang, Guandong Xu, Xindong Wu, and Zhaohui Wu. On deep learning for trust-aware recommendations in social networks. *IEEE transactions on neural networks and learning systems*, 28(5):1164–1177, 2017.

[FBZ14] Hui Fang, Yang Bao, and Jie Zhang. Leveraging decomposed trust in probabilistic matrix factorization for effective recommendation. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[GZTYS14] G. Guo, J. Zhang, D. Thalmann, and N. Yorke-Smith. Etaf: An extended trust antecedents framework for trust prediction. In *Proceedings of the 2014 International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 540–547, 2014.

[GZYS13] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. A novel bayesian similarity measure for recommender systems. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[GZYS15a] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems. *Knowledge-Based Systems*, 74:14–27, 2015.

[GZYS15b] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. Trustsvd: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[HCZ04] Zan Huang, Hsinchun Chen, and Daniel Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):116–142, 2004.

[JE09] Mohsen Jamali and Martin Ester. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 397–406. ACM, 2009.

[JE10] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 135–142. ACM, 2010.

[KKK17] Saakshi Kapoor, Vishal Kapoor, and Rohit Kumar. A review of attacks and its detection attributes on collaborative recommender systems. *International Journal of Advanced Research in Computer Science*, 8(7), 2017.

[Lap12] Theodoros Lappas. Fake reviews: The malicious perspective. In *International Conference on Application of Natural Language to Information Systems*, pages 23–34. Springer, 2012.

[LKH14] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073, 2014.

[LLZ+19] Yezheng Liu, Zhe Li, Chong Zhou, Yuanchun Jiang, Jianshan Sun, Meng Wang, and Xiangnan He. Generative adversarial active learning for unsupervised outlier detection. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[LM16] Wei-Po Lee and Chuan-Yuan Ma. Enhancing collaborative recommendation performance by combining user preference and trust-distrust propagation in social networks. *Knowledge-Based Systems*, 106:125–134, 2016.

[LMY+12] Linyuan Lü, Matúš Medo, Chi Ho Yeung, Yi-Cheng Zhang, Zi-Ke Zhang, and Tao Zhou. Recommender systems. *Physics reports*, 519(1):1–49, 2012.

[MKL09] Hao Ma, Irwin King, and Michael R Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 203–210. ACM, 2009.

[MYLK08] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 931–940. ACM, 2008.

[MZL+11] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296. ACM, 2011.

[NL16] Trong T Nguyen and Hady W Lauw. Representation learning for homophilic preferences. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 317–324. ACM, 2016.

[PHY17] Yiteng Pan, Fazhi He, and Haiping Yu. Trust-aware collaborative denoising auto-encoder for top-n recommendation. *arXiv preprint arXiv:1703.01760*, 2017.

[RC17] Dimitrios Rafailidis and Fabio Crestani. Learning to rank with trust and distrust in recommender systems. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 5–13. ACM, 2017.

[RHW+88] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[RRS11] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.

[SH12] Nikita Spirin and Jiawei Han. Survey on web spam detection: principles and algorithms. *ACM SIGKDD explorations newsletter*, 13(2):50–64, 2012.

[Smo86] Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado Univ at Boulder Dept of Computer Science, 1986.

[SMSX15] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*, pages 111–112. ACM, 2015.

[THGL13] Jiliang Tang, Xia Hu, Huiji Gao, and Huan Liu. Exploiting local and global social context for recommendation. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[WDZE16] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 153–162. ACM, 2016.

[WSY15] Hao Wang, Xingjian Shi, and Dit-Yan Yeung. Relational stacked denoising autoencoder for tag recommendation. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[WWY15] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1235–1244. ACM, 2015.

[YHHZ14] Weilong Yao, Jing He, Guangyan Huang, and Yanchun Zhang. Modeling dual role preferences for trust-aware recommendation. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 975–978. ACM, 2014.

[YLLL16] Bo Yang, Yu Lei, Jiming Liu, and Wenjie Li. Social collaborative filtering by trust. *IEEE transactions on pattern analysis and machine intelligence*, 39(8):1633–1647, 2016.

[YSL12] Xiwang Yang, Harald Steck, and Yong Liu. Circle-based recommendation in online social networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1267–1275. ACM, 2012.

[ZYST19] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1):5, 2019.