A Unified Framework for Reputation Estimation in Online Rating Systems

Guang Ling[1,2], **Irwin King**[1,2], **Michael R. Lyu**[1,2]
[1] Shenzhen Research Institute, The Chinese University of Hong Kong, Shenzhen, China
[2] Department of Computer Science and Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
{gling,king,lyu}@cse.cuhk.edu.hk

Abstract

Online rating systems are now ubiquitous due to the success of recommender systems. In such systems, users are allowed to rate the items (movies, songs, commodities) in a predefined range of values. The ratings collected can be used to infer users' preferences as well as items' intrinsic features, which are then matched to perform personalized recommendation. Most previous work focuses on improving the prediction accuracy or ranking capability. Little attention has been paid to the problem of spammers or low-reputed users in such systems. Spammers contaminate the rating system by assigning unreasonable scores to items, which may affect the accuracy of a recommender system. There are evidences supporting the existence of spammers in online rating systems. Reputation estimation methods can be employed to keep track of users' reputation and detect spammers in such systems. In this paper, we propose a unified framework for computing the reputation score of a user, given only users' ratings on items. We show that previously proposed reputation estimation methods can be captured as special cases of our framework. We propose a new low-rank matrix factorization based reputation estimation method and demonstrate its superior discrimination ability.

1 Introduction

With the growth of online social networks and e-commerce website, more and more rating systems emerged and began to play an important role on today's Internet. In such rating systems, a user is allowed to assign scores to items and indicates her preferences. In a rating system with rich features, additional information such as textual comments can also be provided. Based on these ratings, a recommender system can predict a user's preferences over a set of items. The rating systems running at large service providers like Amazon and Yahoo have tens of millions of users [Bertin-Mahieux *et al.*, 2011]. Thus a small change in recommendation accuracy could affect the sales and profit of the website significantly. Large real-life recommender systems nowadays usually adopt collaborative filtering techniques [Bennett and Lanning, 2007; Bertin-Mahieux *et al.*, 2011]. These techniques are essentially social systems that predict a user's preference on an item by leveraging the judgement of other people. They are vulnerable to spammers and manipulations of ratings. There are evidences showing the existence of spammers in online rating systems. The ratings assigned by spammers can contaminate the system and affect recommendation accuracy. Detecting these spammers and eliminating their ratings from the system is thus crucial to online rating systems.

Malicious users have been found in rating systems in the literature [Gunes *et al.*, 2012]. As early as 2005, there are news reporting on a group of people managing to trick Amazon's recommender system[1]. Recently, there is an incident that a user created non-existing item receiving hundreds of reviews in a major social rating website. Though the item is deleted quickly, this incident still caused a splash in the users. The users' behavior on the fantasy item hinted that their ratings for other items could also be imaginary and malicious. From a recommender system's point of view, such ratings do not have positive impact on the system, because users' irresponsible ratings may contaminate the features of the items that they have rated.

A user reputation system can be employed to detect and mitigate the effect of spammers in such rating systems. A reputation system can estimate the credibility of a user by analyzing the rating behavior of the user. In previous studies of user reputation estimation, an item is usually assumed to have an intrinsic utility that is common to all users [Li *et al.*, 2012; de Kerchove and Dooren, 2010]. Then a user's rating is compared with this utility and the deviation between the rating and the utility determines the reputation score of the user. This quality view of item might be true for some rating systems, for example, an online commodity shop where the ratings are assigned based on the utility of the commodity. However, it may not be true in other types of rating systems. For example in a movie recommendation system, Star Trek might be the favorite movie for a Sci-Fi fan while it is tiring to a documentary lover. The utility score of an item can be quite different for different users in this scenario. Lower the documentary lover's reputation score because she assigned a very low score to Star Trek may proven inappropriate. As long as the scores assigned to items are consistent, one's reputation

[1] http://news.cnet.com/2100-1023-976435.html

score should not be penalized.

Reputation score should measure users' *consistency* and *predictability* in assigning the ratings to items. Given a reasonable model that captures the known ratings, the consistency and predictability can be measured with respect to the model prediction. If a user always assigns ratings to items in a way that is surprising and unpredictable, it is likely that he is a spammer. For example, if a user assigns a very high score to the movie Batman Dark Knight and a very low score to The Dark Knight Rises, it is inconsistent and unpredictable since these two movies are produced by the same director and have very similar characteristics. From the model's point of view, the two ratings are contradictory and cannot be trusted completely. We incorporate this idea into our framework and introduce a new method based on the framework.

The contribution of this work is three-fold. First, we propose a unified framework for reputation estimation in which the fruitful models in collaborative filtering (CF) can be readily plugged-in. We estimate the reputation of a user based on the deviation of her ratings versus the model output and penalize the deviation through a penalty function. Then this penalization quantity is transformed to the reputation through a link function. We show that previously proposed reputation estimation methods can be instantiated as special cases of our framework by choosing suitable CF models, penalize functions and link functions. Secondly, we introduce a low-rank matrix factorization based reputation estimation method by leveraging our framework. Thirdly, we conduct empirical study of various algorithms under several spamming strategies, i.e., *Random attacks*, *Bandwagon attacks* [Burke *et al.*, 2006], *nuke* and *push* [O'Mahony *et al.*, 2004].

The rest of the paper is organized as follows. We review related work in Section 2. In Section 3, we propose the framework and show how previous work can be incorporated into this framework. Section 4 introduces a low-rank matrix factorization based reputation estimation method. We perform empirical study in Section 5 and conclude the paper in Section 6.

2 Related Work

We briefly review related work in collaborative filtering and reputation estimation systems.

2.1 Collaborative Filtering

Collaborative filtering (CF) is a major method to approach recommender systems. The study of CF starts with memory-based methods. These methods include a user-centric view [Breese *et al.*, 1998] or item-centric view [Sarwar *et al.*, 2001]. Later, model-based methods emerged. In a model-based method, a succinct model that explains the observed ratings is built from existing ratings. Through such a procedure, only the model parameters are needed to make predictions of a user's rating on an unseen item. Famous model-based methods include Probabilistic Latent Semantic Analysis (PLSA) [Hofmann, 2001], low rank matrix factorization based methods [Salakhutdinov and Mnih, 2007; 2008; Srebro *et al.*, 2005], etc.. In low rank matrix factorization based methods, users and items are assumed to share a common low rank space. Each user and each item is represented as a vector in this space and the rating that a user assigns to an item is the dot product of their feature vectors. Low rank matrix factorization turns out to be very efficient and attains good results in practice. Recently, there are CF models that try to incorporate other information to improve accuracy, such as social information [Ma *et al.*, 2009; 2011] and time series information [Koren, 2009]. Some ranking oriented methods are also proposed [Koren and Sill, 2011; Shi *et al.*, 2012].

2.2 Reputation Estimation

There are two different scenarios where reputation estimation methods come into play. In one line of research, the trustiness of a user and the propagation of the trust are studied [Guha *et al.*, 2004; Overgoor *et al.*, 2012]. The reputation of a user is calculated based on the trustiness of the user. Such reputation estimation methods are suitable for a graph based system like social networks. Another line of study that operates on rating systems is more relevant to our work. Usually an item is assumed to have an intrinsic quality that is common to all users and users' reputations are calculated based on the deviation between their ratings and the true qualities of the rated items [Mizzaro, 2003; Laureti *et al.*, 2006]. Recently, there are investigations that guarantee the theoretical convergence of the reputations [de Kerchove and Dooren, 2010; Li *et al.*, 2012]. These algorithms are not applicable to every rating system since the intrinsic quality assumption might not be true in all cases. Principle Component Analysis (PCA) and PLSA based methods are proposed in [Mehta *et al.*, 2007a; Mehta and Nejdl, 2009], which exploit statistical properties to identify spam users in a rating system. PCA learns the low rank structure of the data. However, since it sticks to the principle components, the flexibility of the model is questionable. A reputation weighted strategy is adopted in [Mehta *et al.*, 2007b] to improve the robustness of prediction accuracy.

3 Reputation Estimation Framework

In this section, we introduce the reputation estimation framework and demonstrate its adaptability.

3.1 Problem formulation

Given a set of N users $\mathcal{U} = \{u_1, u_2, \cdots, u_N\}$ and a set of M items $\mathcal{I} = \{i_1, i_2, \cdots, i_M\}$, users' ratings on items form an $N \times M$ matrix R, where the entry on the ith row jth column r_{ij} denotes u_i's rating on i_j. Alternatively, we denote users' ratings on items by a set of triples $\mathcal{Q} = \{(i, j, r)\}$, where $r_{ij} \in R$. Let the set of items that are rated by u_i be $\mathcal{I}_i$ and the set of users who have rated i_j be $\mathcal{U}_j$. We assume that the ratings are integers from 1 to D (real ratings can also be accommodated without a problem). Then a reputation estimation model tries to estimate the reputation c_i for each of the users u_i. It is convenient to require that $0 \leq c_i \leq 1$, with a larger value indicating the reputation of u_i is higher.

3.2 The Framework

Our framework is composed of three ingredients as an integrated paradigm, namely the prediction model, the penalty function and the link function.

Prediction Model

Given a collaborative filtering model $\mathcal{H}$ which can predict the entries of R, we assume that the ith row, jth column of R, r_{ij}, is a Gaussian random variable centered at model prediction $\mathcal{H}(i,j)$ with variance σ^2,

$$r_{ij} \sim \mathcal{N}(\mathcal{H}(i,j), \sigma^2).$$

Then the log-likelihood of observing r_{ij} given $\mathcal{H}(i,j)$ is

$$\begin{aligned}\mathcal{L}_{ij} &= \log(P(r_{ij}|\mathcal{H}(i,j))) \\ &= \log(\mathcal{N}(r_{ij}|\mathcal{H}(i,j), \sigma^2)) \\ &= C - \frac{1}{2\sigma^2}(r_{ij} - \mathcal{H}(i,j))^2,\end{aligned}$$

where C is a constant. The quantity

$$s_{ij} = (r_{ij} - \mathcal{H}(i,j))^2$$

can be interpreted as the *unexpectedness*, which is inversely related to the *predictability* of the rating.

If we take the probability mass function $\mathcal{N}(x|\mu, \sigma^2)$ of a Gaussian random variable as the approximation of the probability that an event E happens within a small ϵ-neighborhood of x, then $\mathcal{L}_{ij}$ has a natural explanation, i.e., it is the *self-information* of the outcome E, which is a measure of information associated with the outcome of a random variable. The larger the s_{ij} is, the more surprising it is for the rating r_{ij} is not expected by $\mathcal{H}$. A well behaved user should have small s_{ij} for the ratings. The underlying reasoning is the same as the assumption made by a collaborative filtering methods, i.e., there should be a few factors that can explain the behavior of a normal user. A well-trained model $\mathcal{H}$ should be able to learn the underpinning factors and make reasonable predictions. On the contrary, if the unexpectedness s_{ij} is large for the ratings given by u_i, it is a sign that the ratings are not *accordant* with other ratings in the system and the user is not behaving predictably as the model $\mathcal{H}$ assumes.

If we take a different perspective, we see that s_{ij} is the distance between the model $\mathcal{H}$'s prediction and the observation, which is the L_2 measure of training error. In other words, s_{ij} indicates how well the ratings r_{ij} can be trained using current model $\mathcal{H}$. This method of detecting unexpectedness coincides with the Type 1 approach of outlier detection [Hodge and Austin, 2004].

Since the model $\mathcal{H}$ can predict every entry of R, we can readily compute the unexpectedness of all the known ratings s_{ij}, for all $(i,j,r) \in \mathcal{Q}$. For user u_i, the set $\{s_{ij}|j \in \mathcal{I}_{u_i}\}$ measures the unexpectedness of seeing each of the ratings assigned by the user.

Penalty Function

The purpose of the penalty function is summarizing the set of unexpectedness $\{s_{ij}\}$ into one quantity s_i or s_j, where the former represents the overall unexpectedness of u_i and the later represents the overall unexpectedness of i_j. Then s_i or s_j is used by the link function to get the reputation score for user u_i. As an example, the penalty function that computes s_i can be a simple arithmetic mean,

$$s_i = \frac{1}{\|\mathcal{I}_i\|} \sum_{j \in \mathcal{I}_i} s_{ij}. \tag{1}$$

Link Function

After getting a summarizing unexpectedness s_i for u_i or s_j for i_j, we can compute the reputation score for u_i. This task is performed by a link function. The link function should relate the reputation of a user inversely to the unexpectedness s_i. It is convenient and natural to require that the reputation of a user c_i to lie in between $[0,1]$, a simple link function which fulfills this requirement is

$$c_i = 1 - \frac{s_i}{s_{max}},$$

where s_{max} is the maximum possible value of s_i.

3.3 Adaptability of the framework

We show the adaptability of the proposed framework by showing how previously proposed reputation estimation methods can be captured using suitable $\mathcal{H}$'s, penalty functions and link functions.

Mizzaro's algorithm [Mizzaro, 2003] is one of the earliest work on reputation estimation. It assumes that each item has an intrinsic quality q_j and measures the *steadiness* of object quality, which is then used to determine the reputation of the user. It can be captured by our framework by choosing

$$\mathcal{H}(i,j) = \sum_{i \in \mathcal{U}_j} c_i r_{ij} / \sum_{i \in \mathcal{U}_j} c_i, \tag{2}$$

$$s_j = \sum_{i \in \mathcal{U}_j} c_i,$$

and

$$c_i = \frac{\sum_{j \in \mathcal{I}_i} s_j (1 - \sqrt{\sqrt{s_{ij}}/s_{max}})}{\sum_{j \in \mathcal{I}_j} s_j}.$$

Note that this algorithm is iterative in nature. The employed model predicts u_i's rating on i_j as a weighted average of all the ratings assigned to item i_j, where the weight carried by r_{ij} is the reputation score c_i of a user. The algorithm begins by assigning equal reputation score to all users. Then it iteratively updates c_i and $\mathcal{H}(i,j)$ using current parameter set, until a predefined convergence condition is met.

The algorithm proposed by Laureti et al. [Laureti *et al.*, 2006] can also be seen as an instantiation of the framework by choosing the model $\mathcal{H}$ as in Eq. (2), the penalty function as

$$s_i = \frac{1}{\|\mathcal{I}_i\|} \sum_{j \in \mathcal{I}_i} s_{ij}, \tag{3}$$

and the link function as

$$c_i = (s_i + \epsilon)^{-\beta}, \tag{4}$$

where ϵ is a small constant to prevent diverging and β is the strength of the penalty applied to user. Note that one property of this algorithm is that c_i does not necessarily lie in the range $[0,1]$. However, since the original purpose of their algorithm is to estimate the intrinsic quality of an item, it does not pose a problem.

De Kerchove et al. [de Kerchove and Dooren, 2010] proposed an algorithm that is similar to Laureti's algorithm. It is an instance of the proposed framework by choosing the model

Table 1: Choice of $\mathcal{H}$, penalty function and link function in Li's algorithms

	$\mathcal{H}$	Penalty Function	Link Function
L1-AVG	Eq. (6)	$\frac{1}{\|\mathcal{I}_i\|}\sum_{j\in\mathcal{I}_i}\sqrt{s_{ij}}$	$1-\lambda s_i$
L2-AVG	Eq. (6)	$\frac{1}{\|\mathcal{I}_i\|}\sum_{j\in\mathcal{I}_i} s_{ij}$	$1-\frac{\lambda}{2}s_i$
L1-MAX	Eq. (6)	$\max_{j\in\mathcal{I}_i}\sqrt{s_{ij}}$	$1-\lambda s_i$
L2-MAX	Eq. (6)	$\max_{j\in\mathcal{I}_i} s_{ij}$	$1-\frac{\lambda}{2}s_i$
L1-MIN	Eq. (6)	$\min_{j\in\mathcal{I}_i}\sqrt{s_{ij}}$	$1-\lambda s_i$
L2-MIN	Eq. (6)	$\min_{j\in\mathcal{I}_i} s_{ij}$	$1-\frac{\lambda}{2}s_i$

as in Eq. (2), the penalty function as in Eq. (3) and the link function

$$c_i = 1 - k \times s_i, \tag{5}$$

where k is chosen such that $c_i \geq 0$.

Motivated by the lack of a theoretical convergence guarantee, Li et al. [Li *et al.*, 2012] proposed a class of six algorithms for computing the reputation of users in a rating system. They proved their algorithms converge to a unique solution theoretically, which is a desired property of reputation estimation algorithms. However, in practice, the convergence is not a problem for all the mentioned algorithms [Medo and Wakeling, 2010]. Table 1 summarizes the choices of $\mathcal{H}$, the penalty function and the link function to recover the six algorithms.

$$\mathcal{H}(i,j) = \frac{1}{\|\mathcal{U}_j\|}\sum_{i\in\mathcal{U}_j} r_{ij}c_i. \tag{6}$$

Note that in Eq. (6), the prediction model is slightly different from that in Eq. (2). The denominator in Eq. (6) chosen so as to guarantee the convergence of the algorithms. Also note that the algorithms assume that all the ratings have been transformed into the range $[0, 1]$, which could easily be achieved by mapping $r'_{ij} = (r_{ij} - 1)/(D - 1)$. Through this mapping, the link function is guaranteed to output the reputation c_i in the range of $[0, 1]$. Similarly, these six algorithms are iterative methods with initial c_i set to 1.

The spammer detection algorithm proposed in [Mehta *et al.*, 2007a; Mehta and Nejdl, 2009] can also be captured by choosing $\mathcal{H}$ to be PCA and PLSA, respectively.

4 Reputation Estimation using Low-Rank Matrix Factorization

In Section 3.3 we have shown how can previous methods be taken as instances of our framework. It is interesting to note that they share the same prediction model $\mathcal{H}$ (a slightly different weighting scheme is used in Li's algorithm). The prediction model is an item centric model, i.e., it outputs the same prediction score for all users' ratings on the same item. This prediction score is interpreted as the intrinsic item quality which measures the utility of the item. In the aforementioned methods, the reputation of a user is measured based on how her ratings deviate from the qualities of the items. Such a scheme naturally assumes that there is indeed an intrinsic quality for an item.

However, as we argued in Section 1, the utility of an item could vary depending on the taste of the user. In fact, both the taste view and the intrinsic quality view have their area of application. Different prediction models suit different scenarios. In an e-commerce website, the items are mainly the products that are on sale in the website. In this case, the rating that a user gives to an item should depend more on the quality and functionality of the item. Therefore, the assumption that an item has an intrinsic quality is suitable. On the other hand, in an online stream service website, where music and movies can be purchased and rated, the utility of an item (a song or a movie) could be different depending on the user's taste and preference.

We propose a reputation estimation system that utilizes a personalized prediction model to better capture the case where the taste view is more suitable. The prediction model we use is a low-rank matrix factorization method, which is shown to be a flexible model with good prediction accuracy [Salakhutdinov and Mnih, 2007].

4.1 Low-Rank Matrix Factorization

In a low-rank matrix factorization (MF) method, the rating matrix $R \in \mathbb{R}^{N\times M}$ is assumed to have a low-rank structure, i.e., it has a rank of $K \ll \min\{M, N\}$. Then R can be decomposed into two rank-K matrix $U \in \mathbb{R}^{K\times N}$ and $V \in \mathbb{R}^{K\times M}$ as $R = U^T V$. The column vectors of U and V have a natural interpretation. The ith column vector U_i of U is the latent factor that determines the behavior of u_i and the jth column vector V_j of V is the latent factor that determines the features of i_j. The dot product $U_i^T V_j$ is the model predicted score of u_i's rating on i_j.

The Probabilistic Matrix Factorization (PMF) adopts a probabilistic linear model with Gaussian observation noise [Salakhutdinov and Mnih, 2007]. Maximizing the posterior probability is equivalent to minimizing the following objective function

$$\mathcal{L} = \frac{1}{2}\sum_{(i,j,r)\in\mathcal{Q}} (r - U_i^T V_j)^2 + \frac{\lambda_U}{2}\|U\|_F^2 + \frac{\lambda_V}{2}\|V\|_F^2,$$

where the first term is the squared loss and the last two terms are regularization imposed to avoid over-fitting problem. To minimize the objective function, first compute the gradients of the objective function with respect to U_i and V_j as

$$\frac{\partial\mathcal{L}}{\partial U_i} = \sum_{j\in\mathcal{I}_i}(U_i^T V_j - r_{ij})V_j + \lambda_U U_i,$$
$$\frac{\partial\mathcal{L}}{\partial V_j} = \sum_{i\in\mathcal{U}_j}(U_i^T V_j - r_{ij})U_i + \lambda_V V_j.$$

Then alternative update on U_i and V_j can be done through

$$U_i \leftarrow U_i - \eta\frac{\partial\mathcal{L}}{\partial U_i} \qquad V_j \leftarrow V_j - \eta\frac{\partial\mathcal{L}}{\partial V_j}.$$

After training, the prediction of user i's rating on item j as predicted by the model is the dot product $U_i^T V_j$. In practice, r_{ij} is usually mapped into the range $[0, 1]$ and the sigmoid function is used to filter $U_i^T V_j$ before it is matched to r_{ij}. Apart from the batch-trained algorithm, online algorithms for learning PMF are also proposed [Ling *et al.*, 2012], in which new users and new items can be accommodated more easily.

4.2 Penalty Function and Link Function

The penalty function we choose is the arithmetic mean in Eq. (1) and the link function we employ is

$$c_i = 1 - s_i.$$

We assume that the collected ratings have been mapped to the range $[0, 1]$ so that s_i and c_i are all in the range $[0, 1]$.

5 Experiments

In this section, we conduct empirical studies on the reputation estimation algorithms. We compare the algorithms' discrimination capabilities on identifying spammers in a rating system.

5.1 Datasets

Labeling spam users in a rating system is a tedious and time consuming job. Even for human annotators, usually additional information like text comments is needed to identify a spam user. To the best of our knowledge, there is no publicly available benchmark rating dataset that has ground truth spam user labels. However, there are several rating datasets that are well studied in the collaborative filtering community. These datasets contain ratings collected from real recommender systems. Usually the data in such datasets have gone through a filtering process, i.e., filtering out users who assign too few ratings and items which receive too few ratings. These datasets provide us a chance to evaluate the reputation estimation algorithms. We use MovieLens[2] dataset as the base dataset to evaluate the algorithms.

Basic statistics of MovieLens is shown in Table 2. By today's standard, a million-sized dataset is at most a median-sized dataset. We choose to use this dataset because the data are collected from a non-commercial recommender system. It is more likely that the users in this dataset are non-spam users. We take the users already in the dataset as normal users. We add spam users by mimic their rating behavior through the following strategies.

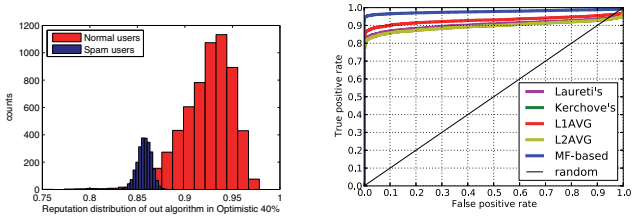
- **Random spamming:** Spam users rate the items in a random way. The rating that a spam user assigns to an item is chosen uniformly at random from 1 to 5. This strategy resembles the *random attacks* seen in spammer attacks [Burke *et al.*, 2006].
- **Semi-random spamming:** Spam users rate half of their ratings randomly as is in the random rating scheme, and the other half rating is copied randomly from normal users' rating on the same item. This is similar to the *average attacks*.
- **Optimistic spamming:** Spam users assign half of their ratings using the highest possible score (5 in our case) and copy randomly from normal users on the other half items. This strategy resembles the *Bandwagon attacks*.
- **Pessimistic spamming:** Spam users assign half of their ratings using the lowest possible score (1 in our case) and copy randomly from normal users on the other half items. This is the *nuke* strategy used by spammers [O'Mahony *et al.*, 2004].

[2]http://www.cs.umn.edu/Research/GroupLens

Table 2: Statistics of MovieLens Dataset

# of ratings	# of users	# of items	rating range
1,000,209	6040	3706	1 to 5

Figure 1: Reputation distribution and ROC curve



(a) Reputation distribution (b) ROC under Optimistic 20%

The number of spam users to add may also impact different algorithms' discrimination abilities. We choose to add 4 different levels of spam users to the dataset, 10%, 20%, 30% and 40% (percentage of spammers to normal users), to investigate the effect of the spammer rate. Each spam user randomly selects the average number of ratings received from normal users to rate.

5.2 Evaluation Methods

To evaluate the performance of various methods, we employ the Area under the ROC Curve (AUC) [Bradley, 1997] to measure their discrimination abilities. A sample distribution of reputation is shown in Figure 2(a). As a vertical separation line swipes through the x-axis, the true positive rate (TPR) and false positive rate (FPR) would vary depending on the position. The trade-off of TPR and FPR is shown in the ROC curve in Figure 2(b). AUC is the area under the ROC curve, which is a good single number summarization of the discrimination ability using only the reputation scores produced by each algorithm. Note the higher the AUC is, the better.

5.3 Implementation Details

For Laureti's algorithm, the β in Eq. (4) is set to -1, which yields superior performance over other choices [Medo and Wakeling, 2010]. Parameter k in Eq. (5) of De Kerchove's algorithm is set to be $k = [\epsilon + \max_{i \in \mathcal{U}} s_i]^{-1}$ to allow maximum separation. For Li's algorithm, λ is set to 1, which yields the best performance. All algorithms are iterated until the change in reputation is less than 10^{-6} for all users.

For our proposed method, we use $K = 10$ latent factors in the comparison. In our experience, the MF model converges very fast and we train the model using 50 iterations.

5.4 Results

Shown in Table 3 are the AURs for different methods under various scenarios. It is evident that our MF-based method outperforms all the compared algorithms under all settings. Especially in the Optimistic case, our method outperforms the runner-up method by about 6% under the 40% spam users

Table 3: Area Under the ROC Curve comparison under various scenarios

Type	**Random**				**Semi-random**				**Optimistic**				**Pessimistic**			
Percentage	10	20	30	40	10	20	30	40	10	20	30	40	10	20	30	40
Laureti's	.9806	.9803	.9797	.9790	.9241	.9240	.9248	.9248	.9464	.9298	.9166	.9047	.9926	.9914	.9902	.9887
Kerchove's	.9793	.9791	.9785	.9777	.9227	.9231	.9239	.9239	.9428	.9234	.9090	.8960	.9910	.9885	.9858	.9829
L1-AVG	.9791	.9789	.9780	.9769	.9098	.9111	.9118	.9115	.9578	.9465	.9376	.9295	.9900	.9875	.9847	.9817
L2-AVG	.9790	.9788	.9782	.9773	.9224	.9228	.9237	.9237	.9425	.9231	.9088	.8959	.9902	.9873	.9841	.9807
MF-based	**.9893**	**.9896**	**.9896**	**.9892**	**.9685**	**.9676**	**.9673**	**.9668**	**.9884**	**.9858**	**.9814**	**.9774**	**.9939**	**.9938**	**.9937**	**.9936**

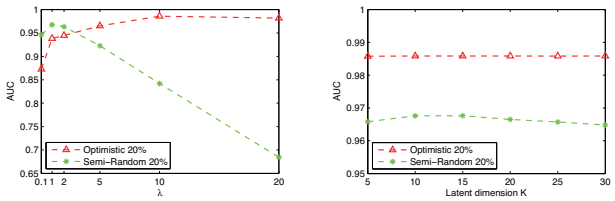
case. In real systems, spam users try to blend into the normal user. They might achieve this by mimic other users' rating on the irrelevant items. And for the items they want to promote, high rating scores should be assigned. Optimistic strategy is a good mechanism for generating such spam ratings. This improvement on AUC means more spam users can be identified without deteriorating the false positive rate. For Li's algorithm, we only show the result of L1-AVG and L2-AVG. Other algorithms including L1-MAX L2-MAX L1-MIN L2-MIN are significantly worse than using the average based penalty function under our experiment setting and we omit the results to save space.

With the increase of spam users percentage, the discrimination ability (measured by AUC) generally decreases, especially under Optimistic and Pessimistic strategies. It is interesting to point out that our method has the least decrease in AUC compared with other baseline methods. This is probably due to the fact that our method utilizes a personalized prediction model while all other methods employ the weighted average. With more spam users, the prediction ability of the weighted average model deteriorates while MF can preserve the meaningful information provided by normal users.

5.5 Sensitivity of Parameters

We study the effect of regularization parameter λ and latent factor number K.

Figure 2: Impact of parameters



(a) Regularization (b) Latent factor number

Regularization

The regularization parameters λ_U and λ_V affect the performance of our algorithm. We use a single parameter λ for both the parameters. We found the impact of λ is different under different spamming strategies. In Random and Semi-random strategies, the optimal λ is smaller than that in the Optimistic and Pessimistic strategies. As an example, we show the impact of λ under Optimistic and Semi-Random strategies in Figure 3(a). The reason for this phenomenon might be that in the Random and Semi-random strategies, a small λ is need to allow the prediction model to fit the normal users' rating better. While in the Optimistic or Pessimistic strategies, a coarse fitting is needed so that the spam users' ratings will not be fitted. We use $\lambda = 10.0$ in Optimistic and Pessimistic scenarios and $\lambda = 1.0$ in Random and Semi-random cases. Our suggestion for choosing λ in real-life scenario is that it should be set to a relatively large value.

Latent Factor Number

The impact of the latent factor number K is shown in Figure 3(b). As we can see, the effect of K is negligible and we choose $K = 10$ in all experiments.

6 Conclusion

Spammers are a serious problem in today's online rating systems. To identify these spam users, a unified framework for reputation estimation is proposed. Many existing prediction models can be readily plugged-in and employed to estimate reputations. We have shown how previously proposed methods can be captured as special cases of our framework. A MF-based reputation estimation method is developed based on the framework. It relaxes the assumption made by previous methods which might not be suitable for all online rating systems. To study the properties of our methods, we conduct a series of carefully designed experiments. Empirically we show the advantageous performance of our method.

There are several interesting directions worthy of consideration for future study. One direction is to study how other prediction models would work in our framework. The second potential avenue is to study how the calculated reputation scores can be incorporated into a recommender system other than trying to eliminate the spam users completely. For example, to develop a collaborative filtering method that is reputation aware.

Acknowledgments

The work described in this paper was fully supported by the Shenzhen Major Basic Research Program (Project No. JC201104220300A) and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. CUHK413212, CUHK415212).

References

[Bennett and Lanning, 2007] J. Bennett and S. Lanning. The Netflix Prize. In *Proceedings of the KDD Cup Workshop 2007*, pages 3–6. ACM, New York, 2007.

[Bertin-Mahieux *et al.*, 2011] Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *ISMIR*, pages 591–596, 2011.

[Bradley, 1997] Andrew P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.

[Breese *et al.*, 1998] John S. Breese, David Heckerman, and Carl Myers Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *UAI*, pages 43–52, 1998.

[Burke *et al.*, 2006] Robin D. Burke, Bamshad Mobasher, Chad Williams, and Runa Bhaumik. Classification features for attack detection in collaborative recommender systems. In *KDD*, pages 542–547, 2006.

[de Kerchove and Dooren, 2010] Cristobald de Kerchove and Paul Van Dooren. Iterative filtering in reputation systems. *SIAM J. Matrix Analysis Applications*, 31(4):1812–1834, 2010.

[Guha *et al.*, 2004] R. Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 403–412, New York, NY, USA, 2004. ACM.

[Gunes *et al.*, 2012] Ihsan Gunes, Cihan Kaleli, Alper Bilge, and Huseyin Polat. Shilling attacks against recommender systems: a comprehensive survey. *Artificial Intelligence Review*, pages 1–33, 2012.

[Hodge and Austin, 2004] Victoria J. Hodge and Jim Austin. A survey of outlier detection methodologies. *Artif. Intell. Rev.*, 22(2):85–126, 2004.

[Hofmann, 2001] Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Mach. Learn.*, 42:177–196, January 2001.

[Koren and Sill, 2011] Yehuda Koren and Joe Sill. Ordrec: an ordinal model for predicting personalized item rating distributions. In *RecSys*, pages 117–124, 2011.

[Koren, 2009] Yehuda Koren. Collaborative filtering with temporal dynamics. In *KDD*, pages 447–456, 2009.

[Laureti *et al.*, 2006] Paulo Laureti, L. Moret, Yi-Cheng Zhang, and Y.-K. Yu. Information filtering via iterative refinement. *CoRR*, abs/physics/0608166, 2006.

[Li *et al.*, 2012] Rong-Hua Li, Jeffrey Xu Yu, Xin Huang, and Hong Cheng. Robust reputation-based ranking on bipartite rating networks. In *SDM*, pages 612–623, 2012.

[Ling *et al.*, 2012] Guang Ling, Haiqin Yang, Irwin King, and Michael R. Lyu. Online learning for collaborative filtering. In *IJCNN*, pages 1–8, 2012.

[Ma *et al.*, 2009] Hao Ma, Irwin King, and Michael R. Lyu. Learning to recommend with social trust ensemble. In *SIGIR*, pages 203–210, 2009.

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

[Medo and Wakeling, 2010] Matus Medo and Joseph R. Wakeling. The effect of discrete vs. continuous-valued ratings on reputation and ranking systems. *CoRR*, abs/1001.3745, 2010.

[Mehta and Nejdl, 2009] Bhaskar Mehta and Wolfgang Nejdl. Unsupervised strategies for shilling detection and robust collaborative filtering. *User Model. User-Adapt. Interact.*, 19(1-2):65–97, 2009.

[Mehta *et al.*, 2007a] Bhaskar Mehta, Thomas Hofmann, and Peter Fankhauser. Lies and propaganda: detecting spam users in collaborative filtering. In *Proceedings of the 12th international conference on Intelligent user interfaces*, IUI '07, pages 14–21, New York, NY, USA, 2007. ACM.

[Mehta *et al.*, 2007b] Bhaskar Mehta, Thomas Hofmann, and Wolfgang Nejdl. Robust collaborative filtering. In *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, pages 49–56, New York, NY, USA, 2007. ACM.

[Mizzaro, 2003] Stefano Mizzaro. Quality control in scholarly publishing: A new proposal. *JASIST*, 54(11):989–1005, 2003.

[O'Mahony *et al.*, 2004] Michael P. O'Mahony, Neil J. Hurley, Nicholas Kushmerick, and Guenole C. M. Silvestre. Collaborative recommendation: A robustness analysis. *ACM Trans. Internet Techn.*, 4(4):344–377, 2004.

[Overgoor *et al.*, 2012] Jan Overgoor, Ellery Wulczyn, and Christopher Potts. Trust propagation with mixed-effects models. In *ICWSM*, 2012.

[Salakhutdinov and Mnih, 2007] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *NIPS*, 2007.

[Salakhutdinov and Mnih, 2008] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML*, pages 880–887, 2008.

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

[Shi *et al.*, 2012] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Alan Hanjalic, and Nuria Oliver. Tfmap: optimizing map for top-n context-aware recommendation. In *SIGIR*, pages 155–164, 2012.

[Srebro *et al.*, 2005] Nathan Srebro, Jason D. M. Rennie, and Tommi S. Jaakola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems 17*, pages 1329–1336. MIT Press, 2005.