The Thesis Committee for Nikita Maple Sudan
certifies that this is the approved version of the following thesis:

Using Social Network Information in Recommender Systems

Committee:

Joydeep Ghosh, Supervisor

Jason Baldridge

Using Social Network Information in Recommender Systems

by

Nikita Maple Sudan, B.A; B.S.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2011

For my sister, Brinda Sudan.

Acknowledgments

I am grateful to Dr. Joydeep Ghosh for guiding me throughout the course of my thesis. My sincere thanks to Dr. Jason Baldridge for his advice and suggestions. I would also like to thank my friend and colleague, Sanmi Koyejo for his patient help and guidance. I am thankful for the MCD Fellowship for the generous support of my work.

Using Social Network Information in Recommender Systems

Nikita Maple Sudan, M.S.E

The University of Texas at Austin, 2011

Supervisor: Joydeep Ghosh

Recommender Systems are used to select online information relevant to a given user. Traditional (memory based) recommenders explore the user-item rating matrix and make recommendations based on users who have rated similarly or items that have been rated similarly. With the growing popularity of social networks, recommender systems can benefit from combining history of user preferences with information from the social/trust network of users. This thesis explores two techniques of combining user-item rating history with trust network information to make better user-item rating predictions. The first approach (SCOAL [5]) simultaneously co-clusters and learns separate models for each co-cluster. The co-clustering is based on the user features as well as the rating history. This captures the intuition that certain groups of users have similar preferences for certain groups of items. The grouping of certain users is affected by the similarity in the rating behavior and the trust network. The second graph-based label propagation approach (MAD [27]) works in a

transductive setting and propagates ratings of user-item pairs directly on the user social graph. We evaluate both approaches on two large public data-sets from Epinions.com and Flixster.com.

The thesis is amongst the first to explore the role of distrust in rating prediction. Since distrust is not as transitive as trust i.e. an enemy's enemy need not be an enemy or a friend, distrust can't directly replace trust in trust propagation approaches. By using a low dimensional representation of the original trust network in SCOAL, we use distrust as it is and don't propagate it. Using SCOAL, we can pin-point the groups of users and the groups of items that have the same preference model. Both SCOAL and MAD are able to seamlessly integrate side information such as item-subject and item-author information into the trust based rating prediction model.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Recommender Systems are used to select online information relevant to a given user. Traditional systems such as collaborative filtering recommend items to users based on users who have rated similarly or items that have been rated similarly. Sinha et al.[24] found that when comparing book and movie recommendations for Amazon.com, MovieCritic.com, Reel.com, RatingZone etc, friends' recommendations almost always outdid the recommendations by collaborative filtering systems. The social aspects of a recommendation can provide additional insights into the decision making process of an individual.

Trust captures user similarity that is not evident from rating history alone. A collaborative filtering system that uses user and item similarity to make a recommendation can be rigged by introducing a fake user profile which rates items similar to the target user and thereby tries to influence the target user's recommendation. Such a copy-user profile attack received publicity "due to a computer glitch that occurred in February 2004 on the Canadian Amazon site. For several days, the mistake revealed the real names of thousands of people who had posted customer reviews of books under pseudonyms. By analyzing the real names, it became evident that the author of a book had

in fact created many different pseudonyms on the Amazon site and used these to write ravishing reviews about her book and rate it highly."[19] By incorporating trust information in the recommender system, the fake profile which is not trusted by any user, will not be able to influence the target user's recommendation as much.

Using trust information also makes the model more interpretable and allows for the "explainability" of a recommendation. The recommender system can make statements such as "Product A was recommended to you because it was loved by your trusted user C". Allowing users to explicitly state which users they trust and to what degree empowers the user to influence the recommendation being made. Mark Zuckerberg, the CEO of Facebook has supposedly said that "A squirrel dying in front of your house may be more relevant to your interests right now than people dying in Africa". With personalization being carried to an extreme, it might be useful for the user concerned about world affairs in addition to the squirrel in front of his/her house, to trust other users such as news channels etc on Facebook.

The cold start problem refers to the inability to recommend to a new user because the system has no information about his/her preferences. If the new user's social network is known in advance, the recommender system can fall back on information from the social network to come up with a recommendation.

Recommender systems can benefit from combining user-item rating history with information from the social/trust network of a user. We explore two

ways of doing this: (1) simultaneous co-clustering and learning (SCOAL) [5] of the user-item rating matrix with a low rank representation of the social network and (2) modified adsorption (MAD) [27] which propagates ratings of user-item pairs directly on the user social graph. We evaluate both approaches on two large public data-sets from Epinions.com and Flixster.com.

The popularity of social rating networks is growing and Flixster alone boasts of more than 10 million registered users. Considering the large number of users and the diverse range of movies/product reviews, it is very unlikely that all the users have the same preference model over all movies or reviews. It is more likely that certain groups of users have similar preferences for certain groups of items. An item here can be a movie or a product review. For example, most teenaged users might like Twilight while older people might not. The teenaged users are also more likely to friend other teen users as compared to older users. SCOAL [5] takes advantage of this intuition by building local preference models for certain groups of users and items.

Related work in the field has primarily focussed on trust propagation([7], [20], [10]) and Matrix Factorization (MF) based approaches([11]). Since distrust is not as transitive as trust i.e. an enemy's enemy need not be an enemy or a friend, distrust can't directly replace trust in trust propagation approaches. By using a low dimensional representation of the original trust network in SCOAL, we use distrust as it is and don't propagate it.

The MF based approaches lack interpretability since it is not clear what the latent features represent and it is hard to reason about the rationale of a

user. Both SCOAL and MAD are highly interpretable and allow us to reason about the user's prediction on a rating. Using SCOAL, we can pin-point the groups of users and the groups of items that have the same preference model. The co-clusters in SCOAL can be likened to communities of users in a trust network sharing their tastes for particular types of movies or reviews. The feature weights across different co-clusters can be compared to see which features are the most weighted, different and thus most influential in affecting a recommendation.

Both SCOAL and MAD are able to seamlessly integrate side information such as item-subject and item-author information into the trust based rating prediction model. A by-product of MAD is a rating distribution over individual users and items. This could be used to rank users and items based on who are more likely to give or get high ratings. Such information is particularly useful for marketing and ranking prediction problems.

1.1 Related Work

Traditional recommender systems like collaborative filtering approaches ([8],[23],[14]) only use user-item rating history and ignore the social relations among users. Lately, there has been a lot of interest in using social network information in recommender systems.

Trust propagation based approaches are primarily concerned with indirect trust i.e. trust which can be inferred from direct trust relationships using transitivity of users. For example, user u connected to user v also trusts

user v's neighbors but to a lesser extent than u trusts v. Tidal Trust[7] computes the predicted rating for a user u on item i as the weighted average of the ratings by all raters who rated item i and are at a shortest path distance from user u. MoleTrust[20] does the same but considers raters up to a fixed maximum-depth, independent of any specific user or item.

Jamali et al.[10] proposed a random walk method, TrustWalker that combines trust-based and item-based recommendation. Each random walk on the user trust graph returns a predicted rating for user u on target item i. The probability of stopping is directly proportional to the similarity between the target item and the most similar item j, weighted by the sigmoid function of step size k. The more the similarity, the greater the probability of stopping and using the rating on item j as the predicted rating for item i. As the step size increases, the probability of stopping decreases. Thus ratings by closer friends on similar items are considered more reliable than ratings on the target item by friends further away.

The final predicted rating is an aggregate of the ratings from several different random walks. The algorithm stops exploring after six hops on the trust network assuming six degrees of separation. Coverage is defined to be the percentage of test user-item pairs that the system can make a prediction on. TrustWalker shows better coverage and precision than item based collaborative filtering. When compared to Tidal Trust and Mole Trust, TrustWalker shows better coverage and similar precision. However Tidal Trust and Mole Trust show better coverage for cold start users which the authors define to be users

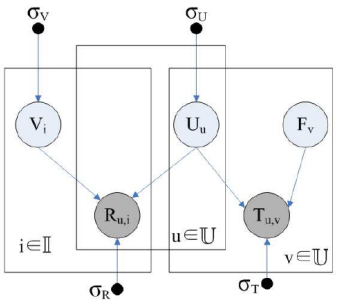

Figure 1.1: Graphical model for Social Recommendation using probabilistic matrix factorization (SOREC)

with less than five ratings.

A limitation of TrustWalker is that unlike SCOAL and MAD, it cannot use available side information for the users, items. Such side information (for e.g. user demographic information, item subject categories etc) can be valuable in predicting a user's rating on an item. MAD provides a probability distribution over all possible rating values which can be used as a confidence measure for the predicted rating. Significance tests on the feature weights for the co-clusters in SCOAL could also be used as confidence measures for the predicted rating.

Since distrust is not as transitive as trust i.e. an enemy's enemy need not be a friend, distrust can't directly replace trust in trust propagation approaches. By using a low dimensional embedding of the original trust network in SCOAL, we use distrust as it is and don't propagate it.

Probabilistic Matrix Factorization (PMF) models the user vs. item rating matrix as a product of two lower-rank user and item matrices. [23] Ma

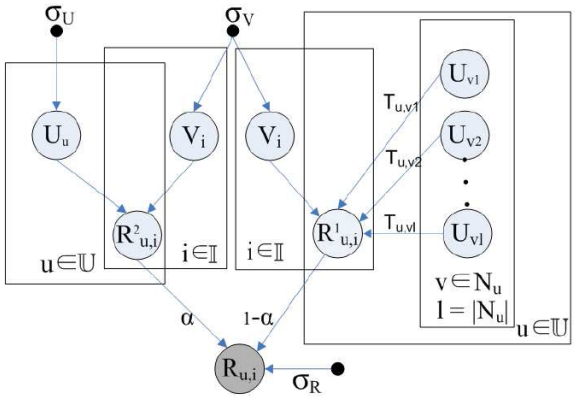

Figure 1.2: Graphical model for Social Trust Ensemble (STE)

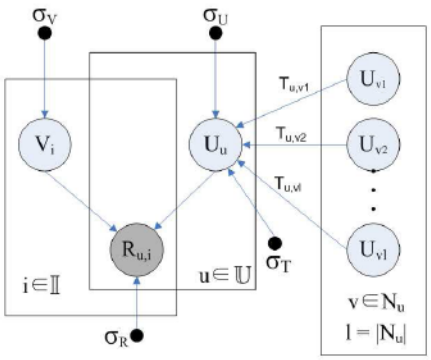

Figure 1.3: Graphical model for Social MF

et al. [18] incorporate the social network graph into such a model by jointly factorizing the rating matrix and the social trust graph. The social network graph is factorized using $U^T F$ and $U^T V$ where the shared low-dimensional matrix U denotes the user latent feature space, Z is the factor matrix in the social network graph and V represents the low-dimensional item feature space. Figure 1.1 shows the graphical model for this approach. The same authors proposed a better model, Social Trust Ensemble (STE) [17] by making the latent features of a user's direct neighbors $(U_{v1}, ..., U_{vl})$ affect the rating of the user. The graphical model for STE is shown in Figure 1.2. STE is shown to outperform existing trust propagation based approaches and Ma et al.'s previous work which factorizes the trust matrix T using latent user features U and latent factor features F (see Figure 1.1). Experimental results are reported on a smaller Epinions dataset different from ours; one that lacks distrust information.

SocialMF[11] is a matrix factorization based model which incorporates social influence by making the features of every user depend on the features of his/her direct neighbors in the social network. The latent features of users indirectly connected in the social network would become dependent by transitivity and hence the trust propagated. The graphical model for this approach can be seen in Figure 1.3. SocialMF showed lower RMSE (on the Epinions, Flixter datasets) as compared to STE. The Epinions dataset that SocialMF was evaluated on is different from ours and lacks distrust information. MF approaches lack interpretability because it is not clear what the latent features

represent. It is also not clear which latent features (user or item) are the most influential.

Jamali et al. [12] analyze the temporal behavior of users in a social rating network. The authors claim that their model is the first to represent all four effects i.e. social relations-on-ratings (social influence), social relations-on-social relations (transitivity), ratings-on-social relations (selection) and ratings-on-ratings (correlational influence). They exclusively focus on network evolution over time with the goal of generating a social rating network that most resembles the dataset (a different Epinions dataset and Flickr). Unlike SCOAL they don't identify and group users who locally share the same preference models perhaps due to similar effects.

1.2 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 describes the datasets, the feature extraction and the experimental setup for evaluation. The thesis is amongst the first in published work to use distrust information for the task of predicting ratings.

Chapter 3 discusses the incorporation of trust network information into the first approach SCOAL [5] which simultaneously co-clusters and learns separate models for each co-cluster. This captures the intuition that certain groups of users have similar preferences for certain groups of items. The grouping of certain users is affected by the similarity in the rating behavior and the trust network dynamics. Using SCOAL, we can pin-point the groups of users and

the groups of items that have the same preference model. Since distrust is not as transitive as trust i.e. an enemy's enemy need not be an enemy or a friend, distrust can't directly replace trust in trust propagation approaches. By using a low dimensional representation of the original trust network in SCOAL, we use distrust as it is and don't propagate it.

Chapter 4 describes the incorporation of trust network information into MAD [27], a graph based label propagation algorithm. The ratings of user-item pairs are directly propagated on the user social graph. Results are presented and compared with those of SCOAL. Both SCOAL and MAD are able to seamlessly integrate side information such as item-subject and item-author information into the trust based rating prediction model. Chapter 5 concludes with directions for future work.

Chapter 2

Data Description and Selection

Due to the sensitive nature of social network data, there are very few publicly available datasets. The extended epinions dataset [19] and Flixster [11] are among the few that are publicly available.

Massa et al [19] collected the extended epinions dataset alongside a smaller epinions dataset to judge the overall trustworthiness of a user. They argue that social networks have a significant number of "controversial users" i.e. users who are trusted by many and distrusted by many. A globally agreed trust value does not exist for such users. They propose local trust metrics that are able to predict the trustworthiness of a user in a personalized way, depending on the personal views of the judging user as opposed to commonly used global trust metrics which assume a unique value of reputation for every single user. Massa et al.[19] however do not report results on the extended epinions dataset and stick to the smaller epinions dataset because it reduces the complexity of their leave-one-out evaluation. The flixster dataset was collected by Jamali et al. for Social MF [11].

Epinions.com is a consumer review site where users share articles/reviews on a variety of products. Users rate a review based on the quality and

usefulness of the review. The site offers a web of trust which is a network of users whose reviews and ratings were found to be consistently valuable. A user can manually add reviewers and raters to his/her web of trust. In addition to trust information, the extended Epinons.com data-set also contains distrust information i.e. the reviewers and raters a user has consistently found to be bad. As far as we are aware, none of the published work uses distrust information to predict the rating of a review.

The epinions data-set also has additional author and subject information for the reviews. The ratings in Epinions are on a scale of 1-5 with a step size of 1. The ratings in Flixster are more granular and can range from 0.5 to 5.0 with a step size of 0.5.

Flixster.com is a social movie site where users can rate movies, discover new movies and meet others who share their taste in movies. Unlike epinions, social relations in flixster are undirected.

Both datasets contain time-stamps for the ratings. The epinions dataset additionally includes time-stamps for the formation of trust relationships. The trust time-stamps range from Jan 10th, 2001 to May 30th, 2002. The Flixster dataset contains users' ratings from Novemeber 2005 to November 2009. Table 2.1 summarizes the number of users, ratings, social relations, etc. for the two datasets. Please see Appendix 1 for snippets of the datasets.

Nguyen et al.[22] used the extended epinions dataset to show that a simple model using derived trustor and trustee features predicted trust rela-

Number	Epinions	Flixster
Ratings	13,668,320	8,196,071
Social Relations	841,372	7,058,819
Users	120,491	147,610
Reviews or Movies	755,760	48,974

Table 2.1: Data statistics for Epinions and Flixster

tions better than Mole Trust [20]. Teng et al.[28] show that publicizing the trust network affects ratings. "The potential to reciprocate produces higher and more correlated ratings than when individuals are unable to see how others rated them." [28]. Epinions.com does allow users to view the people who trust him/her and it is thus worthwhile to include trust information in the rating prediction problem. Most other work ([26],[21]) has used the extended epinions dataset for link prediction and community formation in social rating networks.

2.1 Data Selection

Before attempting SCOAL and MAD on the entire dataset, we first selected a small subset for quick development and comparisons. The small subset was meant to be a testing ground to settle on a range for λ_{ridge} and the dimensionality of the low dimensional representation of the user trust network. To ensure that there were enough ratings per user and per item to make predictions on, we iteratively selected the 4750 most prolific users and the 1000 most popular items from the big (entire) epinions dataset.

However, doing so gave us a very skewed subset with over 90% of the

Rating value	Fraction of knowns with this rating
1	0.0009
2	0.0350
3	0.1853
4	0.1598
5	0.6191

Table 2.2: Distribution of the ratings on a scale of 1 to 5 for the 985 users by 1000 items Epinions subset.

Rating value	Fraction of knowns with this rating
1	0.0002
2	0.0232
3	0.0561
4	0.156
5	0.7695

Table 2.3: Distribution of the ratings on a scale of 1 to 5 for the entire Epinions dataset.

Cross Validation Set	Number of users with less than five ratings	Number of items with less than five ratings
1	265	0
2	271	0
3	284	0
4	276	0
5	283	0

Table 2.4: Statistics for the five-fold cross validation sets on the 978 users by 1000 items Epinions subset

Cross Validation Set	Number of users with less than five ratings	Number of items with less than five ratings
1	75539	250131
2	75442	250227
3	75438	250111
4	75477	250330
5	75549	249782

Table 2.5: Statistics for the five-fold cross validation sets on the entire Epinions dataset

Cross Validation Set	Number of users with less than five ratings	Number of items with less than five ratings
1	84696	25170
2	84506	25226
3	84597	25172
4	84576	25201
5	84579	25106

Table 2.6: Statistics for the five-fold cross validation sets on the entire Flixster dataset

ratings with a value of 5. The iterative approach was thus replaced by selection based on highest variance amongst the ratings weighted by the number of ratings. As seen from Table 2.2, the latter approach gave a more balanced subset which better resembled the skewedness in the entire dataset (Table 2.3). For the flixster dataset, we removed the ratings for users who were not part of the trust network. We also removed the users and items that did not have any ratings associated with them. Tables 2.4-2.6 show the number of users and items that had less than five ratings for all the datasets.

2.2 Feature Extraction and dimensionality reduction

Dimensionality reduction is an interesting alternative to feature selection. Similar to feature selection, it provides a low dimensional representation of the data which can then be used as input for supervised or semi-supervised algorithms. Unlike feature selection, dimensionality reduction preserves information from all the original input variables. In fact, if the data indeed lies on a low-dimensional manifold, it may preserve almost all of the original information while representing it in a way that simplifies learning. Since most dimensionality reduction approaches including the ones we used are purely unsupervised, they may throw away low variance predictions that might be highly predictive of the target label. Dimensionality reduction techniques are also used for visualizing the projected data (two or three dimensions at a time) so as to better understand it. The reader is referred to [3] and [16] for an excellent survey of spectral dimensionality reduction techniques.

Since representing the social or trust network of users as an adjacency matrix would be inefficient (would need $O(numberof - links)$ when stored as sparse), a low dimensional representation of the social network was sought. Reducing the dimensions of the features also helps reduce the computation time for the algorithm and preserve the most variational/distinguishing information from the trust network. Since distrust is not as transitive as trust i.e. an enemy's enemy need not be a friend, distrust should not be propagated. By using a low dimensional embedding of the original trust network in SCOAL, we use distrust information as it is and don't propagate it. Two methods—(i) sparse singular value decomposition (SVD) which is a low rank approximation of a real or complex matrix and (ii) spectral dimensionality reduction using Laplacian Eigenmaps [16] were used to embed the social network in a lower dimension. The dimensionality of the latent factors was taken to be 5 and on concatenating the two latent factors U and V, the dimensions of the social network were reduced from n by n to n by 10. The steps for computing the laplacian eigenmap are detailed as follows.

- Assume that $G = (V, E)$ is an undirected weighted graph with weight matrix W where $w_{ij} = w_{ji} \geq 0$. The adjacency matrix A was made symmetric by doing $W = A + A^T$ and $W_{ii} = 1 \forall i \in n$ based on the assumption that users trusted themselves.

- $W = (w_{ij}) \ \forall \ i, j = 1, ..., n$

- D is a diagonal matrix where $d_{ii} = \sum_{j=1}^{n} w_{ij}$

- The graph laplacian is deinfed as $L = D - W$

The square-roots of the k smallest eigenvectors $u_1, ..., u_k$ of L are used to represent the trust network. We used the user trust adjacency matrix as input for sparse SVD. Laplacian eigenmaps requires the input adjacency matrix to be symmetric whereas SVD does not make this assumption. Since trust relationships are directed, asymmetry is preferred and we stuck to SVD for our experiments on the big/entire datasets.

2.3 Evaluation metric

MSE (mean squared error) is a commonly used metric when comparing recommender systems. It is defined as follows:

$$MSE = \sum_{u,i|R_{test}} (r_{u,i} - \hat{r}_{u,i})^2 / |R_{test}| \tag{2.1}$$

where $r_{u,i}$ is the actual rating by user u on item i, $\hat{r}_{u,i}$ is the predicted rating and R_{test} is the set of all rating pairs in the test set. Another commonly used metric is $RMSE$ (root mean squared error) which is defined as the $\sqrt{MSE}$.

The other commonly used metric is MAE (Mean absolute error) which was used by [20] when doing leave one out evaluation. Leave one out evaluation is computationally inefficient since it requires training on all but one rating and averaging the results over all predictions. Considering the large scale of

our dataset, we restrained from using the leave one out scheme. Extended Epinions dataset had not yet been used for the task of predicting ratings and since Jamali et al. [11] had used MSE as the metric for evaluation on the Flixster dataset, we stuck to our choice of using MSE as the evaluation metric.

We use 5-fold cross validation for training and testing SCOAL. Since MAD is set in a transductive setting, the ratings in the training set are equivalent to the labeled nodes and the algorithm is evaluated based on the MSE computed on the unlabeled rating nodes i.e the ratings in the test set.

Chapter 3

Discriminative approach to using social network information in recommender systems

3.1 SCOAL

The popularity of social rating networks is growing and Flixster alone boasts of more than 10 million registered users. Considering the large number of users and the diverse range of movies/product reviews, it is very unlikely that all the users have the same preference model over all movies or reviews. It is more likely that certain groups of users have similar preferences for certain groups of items. An item here can be a movie or a product review. For example, most teenaged users might like Twilight while older people might not. The teenaged users are also more likely to friend other teen users as compared to older users. SCOAL [5] takes advantage of this intuition by building local preference models for certain groups of users and items.

Collaborative filtering approaches [9] recommend products to users based on rating history alone and ignore user/item features. A classification model on the other hand will form a map between the feature vector for a given user-item pair and the rating but will ignore nearby users or items in the process. A co-clustering approach ([4], [6]) will simultaneously cluster

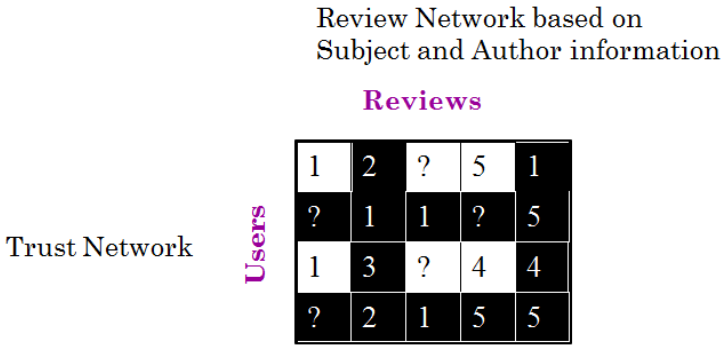

Figure 3.1: User vs. Item (or review) rating matrix.

the users and items based on the ratings. It will then use the ratings of the corresponding co-cluster to predict a missing value. However similar to the collaborative filtering approaches, social network information and additional information such as subject categories will be ignored.

Figure 3.1 shows the matrix of users vs. items where each matrix entry r_{ij} is the rating of a user i on item j. The items can be likened to product reviews in Epinions and movies in Flixster. The ratings in Epinions are on a scale of 1-5 with a step size of 1. The ratings in Flixster are more granular and can range from 0.5 to 5.0 with a step size of 0.5.

SCOAL[5] breaks down the user vs. item rating matrix into (possibly non-contiguous) blocks of rows and columns and builds local models for each block or co-cluster. The darkened region of Figure 3.1 shows an example of a co-cluster consisting of rows 2, 4 and columns 2, 5. Thus users 2 and 4 share the same preference model when rating items 2 and 5. The local models are learnt using the ratings in the co-cluster as well as the row and column

features. One can also specify cell level dyadic features which are features for a particular user-item pair. The row and column membership for each co-cluster with β_{kl} as the model parameters, minimizes the following cost function:

$$\phi_{(SCOAL)} = \sum_{k,l=1}^{K,L} \sum_{\rho(u)=k} \sum_{\gamma(i)=l} w_{ui}(r_{ui} - \beta_{kl}^T x_{ui})^2 \tag{3.1}$$

where K is the number of row clusters, L is the number of column clusters; ρ and γ are the row and column cluster assignment functions; x_{ui} is a concatenated list of row, column and cell level features.

w_{ui} is 1 for known entries and 0 for missing ones. This allows the the missing entries to drop out of the objective function. Since the objective function of SCOAL only depends on the known entries of the matrix, the algorithm is linear in terms of the number of known ratings. SCOAL has shown better performance than collaborative filtering on real life datasets including Movie Lens and Erim.[5]

We incorporate social network information into SCOAL by using the low dimensional representation of the social network as a row or user feature. The user features are thus the concatenated latent features from sparse SVD (Singular Value Decomposition) on the adjacency matrix of the user trust network. A linear regression on the ratings with user and item features is used as the prediction model.

Similar to the users, an item trust network was created using the subject category information. An edge between item i and item j represents that both

items share the same subject. Similar to the user trust features, latent features from sparse SVD on the item-item adjacency matrix were used to represent the network in lower dimensions. The item-subject-author network is the sum of the item-subject network and the item-author network. Thus an edge between item i and j would have a weight of 1 if i and j shared either the same subject or author. The edge weight would be 2 if items i and j had the same author and subject.

The co-clustering is thus dependent on the similarity between ratings and the most representative features of the social/trust network and the item network. If available, additional side information such as user demographics, item descriptions etc could also be used included as user and item features for SCOAL. Instead of restraining from predicting on an item that lacked item information, we zeroed in the item features for such items. The item features being zeroes would simply drop out of the linear regression and the regression in this case would only depend on the user features.

SCOAL does not solve cold-start since it only makes use of the users whose preference ratings are known during training. Assuming, we know the trust information for the cold start users, we can still come up with a prediction for the rating. Semi-supervised SCOAL (SS-SCOAL) [1] is an extension of SCOAL which assigns a new user or item to a co-cluster based on the similarity between the user and item features.

$$\phi_{(SS-SCOAL)} = \phi_{(SCOAL)} + \lambda_r \sum_{k=1}^{K} \sum_{\rho(u)=k} ||v_u - \mu_k||^2 + \lambda_c \sum_{l=1}^{L} \sum_{\gamma(i)=l} ||v_i - \eta_l||^2 \quad (3.2)$$

The original objective function of SCOAL is appended with two additional terms (see equation 3.2) that minimize the squared difference of the user, item features from the mean user feature (μ_k) and mean item feature (η_l) of the co-cluster over all possible row, column assignments. Cold start users and items can thus be assigned to a co-cluster that has the most similar user and item features.

3.2 Results

The earlier experimental results on SCOAL which used information in addition to past ratings were for significantly smaller datasets—(i)Yahoo Movies (7642 users, 11915 movies, 0.23 density), (ii) Movie Lens (943 users, 1682 movies, 0.06 density) and (iii) Erim (1714 households, 121 products, 0.25 density). [1], [5]

The prediction code for SCOAL was vectorized to handle the high number of users and items in Epinions (120491 users, 755760 items) and Flixster (147610 users, 48794 items). The high sparsity in the two datasets (0.0011 density for Epinions and 0.00015 density for Flixster) eased this task and allowed us to speed things up at the cost of memory. Please see Appendix 2 for more details.

As discussed in chapter 2, the dimensions of the social network were reduced using spectral dimensionality reduction and sparse SVD. Ridge regression was used to control model complexity and thereby prevent over-fitting. We tried different values of the L2 regularization parameter, λ_{ridge} and as seen in Table 3.1, the MSE was the lowest for $\lambda_{ridge} = 10^4$ on the 978 users by 1000 items subset from Epinions.

The global regression baseline is equivalent to running SCOAL with $K = 1$ and $L = 1$. The user and item bias baseline was computed as follows:

$$\hat{r_{ui}} = 0.5 * (mean\_row\_bias) + 0.5 * (mean\_col\_bias) + global\_mean$$

where $mean\_row\_bias = \sum_j (R_{uj} - global\_mean)/|j|$ and $mean\_col\_bias = \sum_v (R_{vi} - global\_mean)/|v|$. The $global\_mean$ is the mean of all the known ratings in the training set.

Tables 3.1 and 3.2 only use user trust information (as user features) and ignore item features.[1]

We realized that such high dimensions (of the order of 300, 600 etc) weren't necessary to represent the social network. Dimension sizes of 10, 20, 30 gave similar results. As seen in Table 3.2, for $\lambda_{ridge} = 10^3$, a dimension size of 20 gave the same MSE (i.e. 0.7442) as a dimension size of 600.

[1]The validation for the 978 users by 1000 items was not strictly 5-fold cross-validation because we split all the entries of the user-item data matrix (i.e. both knowns and unknowns) into five sets and then did the routine for holding one set out for test while combining the other four for training. Since during k-fold cross-validation, the splitting of the data is only done on the knowns, this was corrected when preparing the sets for cross-validation on the entire data for both Epinions and Flixster. The SCOAL results in the following chapter use actual 5-fold cross validation on the 978 users and 1000 items.

Spectrally reduced dimension	Global baseline (1,1)	(2,2)	(3,3)	(4,4)
$\lambda_{ridge} = 10^2$				
300	**0.7511**	0.7675	0.7878	0.8035
600	**0.7511**	0.7642	0.7874	0.8035
$\lambda_{ridge} = 10^3$				
300	**0.8168**	0.9779	1.0888	1.2841
600	**0.7442**	0.7478	0.7657	0.7835
$\lambda_{ridge} = 10^4$				
300	0.7442	**0.7223**	0.7295	0.7359
600	0.7442	**0.7213**	0.7308	0.7355

Table 3.1: Mean Squared Error for varying λ_{ridge} and (K, L). MSE averaged over 4 runs for 5-fold cross validation for the 978 users by 1000 items subset from Epinions. The best results are in bold.

Spectrally reduced dimension	Global baseline (1,1)	(2,2)	(3,3)	(4,4)
10	**0.7442**	0.7513	0.7657	0.7835
20	0.7827	0.7448	0.7694	**0.7442**
30	**0.7442**	0.7449	0.7657	0.7815

Table 3.2: Mean Squared Error for varying (K, L) and $\lambda_{ridge} = 10^3$. MSE averaged over 4 runs for 5-fold cross validation [1] for the 978 users by 1000 items subset from Epinions. The best results are in bold.

Having found $\lambda_{ridge} = 10^4$ to give the best results on the small subset (see Table 3.1), we fixed the value of λ_{ridge} as 10^4 for all later experiments.

Table 3.3 compares the MSEs for SCOAL using (i) item features derived from the item-item network based on sharing of a subject and/ author (Item-Subject-Author), (ii) only user features derived from the trust relationships in the user trust network (Trust), (iii) only user features derived from both the trust and distrust relationships in the user trust network (Trust and Distrust), (iv) using features for (i) and (iii) with the modification that the item-item network was only based on sharing of subjects and did not include author information (Trust, Distrust and Item-Subject), (v) using features for (i) and (iii) (i.e. Trust, Distrust and Item-Subject-Author).

Table 3.4 compares the MSEs for SCOAL with SS-SCOAL. Table 3.5 shows the results on the entire Flixster dataset consisting of 147610 users and 48794 items.

3.3 Discussion

SCOAL with $(K, L) = (1, 1)$ is equivalent to having a single global linear regression on the known ratings. Co-clustering seems to help and it beats the global regression baseline for the results on the entire datasets. The MSEs seem to improve as the number of row and column clusters increases. Since Tables 3.1 and 3.2 only use user trust information (as user features) and no item features, the row, column co-cluster assignments are only based on the user trust network and the rating values themselves. In the case of Flixster,

Table 3.3: Mean Squared Error for varying (K, L) and $\lambda_{ridge} = 10^4$. MSE averaged over 4 runs for 5-fold cross validation for the entire Epinions dataset consisting of 120491 users and 755760 items. The best results are in bold.

Predicting the mean	0.4759				
User and Item bias baseline	0.2698 (±0.0006)				
Features	Global baseline (1,1)	(2,2)	(3,3)	(4,4)	(5,5)
Item-Subject-Author	0.4756 (±0.0008)	0.2917 (±0.0014)	0.2478 (±0.0006)	0.2398 (±0.2381)	0.2408 (±0.0028)
Trust	0.4724 (±0.0006)	0.2868 (±0.0013)	0.2460 (±0.0012)	**0.2371** (±0.0008)	0.2399 (±0.0014)
Trust and Distrust	0.4672 (±0.0006)	0.2843 (±0.0012)	0.2446 (±0.0006)	**0.2358** (±0.0002)	0.2379 (±0.0017)
Trust, Distrust and Item-Subject	0.4667 (±0.0007)	0.2860 (±0.0006)	0.2446 (±0.0009)	**0.2370** (±0.0025)	0.2383 (±0.0013)
Trust, Distrust, Item-Subject-Author	0.4667 (±0.0007)	0.2864 (±0.0008)	0.2443 (±0.0012)	**0.2368** (±0.0018)	0.2385 (±0.0017)

Table 3.4: Mean Squared Error for varying (K, L) and $\lambda_{ridge} = 10^4$. MSE averaged over 4 runs for cross-validation set 1 of the entire Epinions dataset consisting of 120491 users and 755760 items. The best results are in bold.

Predicting the mean	0.4759				
User and Item bias baseline	0.2698 ($\pm$0.0006)				
Features	Global Baseline (1,1)	(2,2)	(3,3)	(4,4)	(5,5)
Item-Subject-Author	0.4757	0.2926	0.2476	0.238	0.241
Trust	0.4724	0.2867	0.2453	**0.2369**	0.2391
Trust and Distrust	0.4672	0.2845	0.2446	**0.2358**	0.2379
Trust, Distrust and Item-Subject	0.4667	0.2868	0.2458	**0.2405**	0.2361
Trust, Distrust, Item-Subject-Author	0.4668	0.2872	0.2455	**0.2376**	0.2381
SS-SCOAL using Trust, Distrust and Item-Subject	0.4657	0.4657	0.4646	0.3847	**0.3442**

Predicting the mean	1.1924						
User and Item bias baseline	0.8889 (±0.0009)						
λ_{ridge}	Global Baseline (1,1)	(2,2)	(3,3)	(4,4)	(5,5)	(6,6)	(7,7)
10^4	1.3315 (±0.3155)	0.9601 (±0.0016)	0.8907 (±0.0012)	0.8589 (±0.0020)	0.8481 (±0.0031)	**0.8376** (±0.0022)	0.9131 (±0.0014)
10^5	1.1094 (±0.0005)	0.9567 (±0.0015)	0.8860 (±0.0016)	0.8592 (±0.0016)	0.8472 (±0.0023)	**0.836** (±0.0027)	0.8373 (±0.0033)

Table 3.5: MSE for varying (K, L) with $\lambda_{ridge} = 10^5$. MSE averaged over 4 runs for 5-fold cross validation for the Flixster dataset consisting of 147610 users and 48794 items. The best results are in bold.

this is all we have. Including distrust information (available only for Epinions) in the social network, brings down the MSE from 0.2371 to 0.2358 (see Table 3.3).

As seen in Tables 3.3 and 3.5, SCOAL with trust information outperforms the user and item bias baseline. Table 3.3 shows that SCOAL with trust information does better than SCOAL using only item-subject and item-author information. This shows that user trust information is informative and including it in the prediction model betters performance.

The main advantage of SCOAL over other techniques is its "explainability" in terms of recommendation. We can pin-point the groups of users and the groups of items that have the same preference model. The feature weights across different co-clusters can be compared to see which features were the

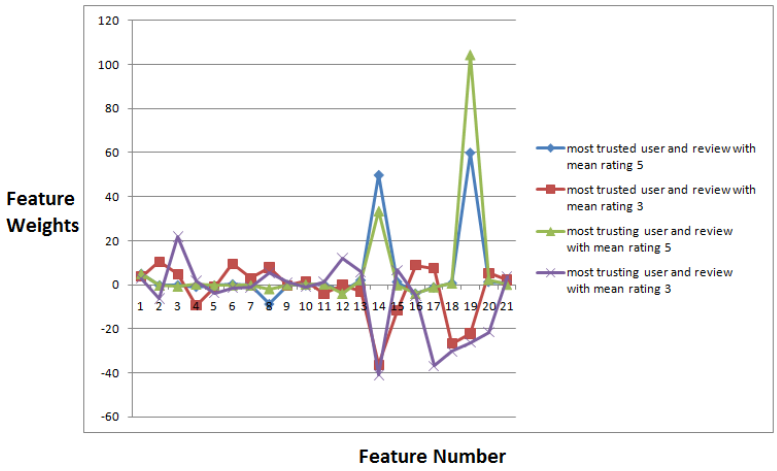

Figure 3.2: Feature weights for the most trusted user and the most trusting user in Epinions when rating a review with a mean rating of 5 and another with a mean rating of 3. Features 2-11 are trust based features whereas features 12-21 are item-subject features.

most weighted, different and thus most influential in affecting a recommendation.

As an example, let's pick two users from the Epinions dataset—(i) the most trusting user i.e. the user who gave the maximum number of $trusts - distrusts$ and (ii) the most trusted user i.e. the user who received the maximum number of $trusts - distrusts$. A trust from user u to user v indicates that user u trusts user v as a rater (of a review) or a reviewer. In the words of Epinions.com, "Your Web of Trust is a network of reviewers whose reviews and ratings you have consistently found to be valuable." Distrust is the opposite of trust. Figure 3.2 plots the feature weights for the most trusted user and the most trusting user when rating a review with a mean rating of 5 and

Row cluster	Number of rows or users
1	38358
2	9178
3	40696
4	32259

Table 3.6: The number of rows in each row cluster.

another review with a mean rating of 3. As can be seen from the figure, both users follow a similar trend for the review features (i.e. features 12-21) when rating a review highly. The same is true for both users rating a review with a medium rating of 3. Both users tend to rely more on the trust features (2-11) when giving a review an average score as compared to a high one. This might be because a really good review might be easier to rate as compared to an average review in which case the user is doubtful on which way to sway the decision. The user might then fall back on the trust network for additional guidance.

The lack of improvement in MSE (see Table 3.3) seemed to indicate that adding item-subject and later item-author information may not lead to better prediction of ratings. However as seen in the example above, additional side information as derived from subjects and authors can still provide a better understanding of the rationale for individual users.

Tables 3.7 and 3.8 show the distribution of the row cluster assignments for the followers and followees of the most trusted and the most trusting user

Assigned row cluster	Percentage of followers assigned to row cluster	Percentage of followees assigned to row cluster
1	11.79	18.64
2	**32.05**	26.06
3	25.90	13.74
4	30.26	**41.56**

Table 3.7: Row cluster assignments for the 1819 followers and 390 followees of the most trusted user in Epinions. The most trusted user was him/herself assigned to row cluster 2.

Assigned row cluster	Percentage of followers assigned to row cluster	Percentage of followees assigned to row cluster
1	22.09	33.33
2	20.46	20.00
3	24.71	8.89
4	**32.74**	**37.78**

Table 3.8: Row cluster assignments for the 3201 followers and 45 followees of the most trusting user in Epinions. The most trusting user was him/herself assigned to row cluster 4.

in Epinions. A follower of user u is a person who trusts user u and a followee of u is a person whom user u trusts. The model learnt by SCOAL assigned the most trusted user to row cluster 2 and the most trusting user to row cluster 4. Table 3.6 shows the sizes of the four row clusters for all users.

The majority of the followers of the most trusting user belong to the same row cluster as the most trusting user. This goes along the intuition that active network users i.e. users who take the time out to trust and distrust others follow the preference model of the people they mostly trust. The followees on the other hand don't seem to care about the preference models of their followers. The majority of the followees of the most trusting user are assigned to a different row cluster than the most trusting user. These users don't seem to rely on the most trusting user's preference model and instead adopt the most trusted user's preference model i.e. row cluster 2. The most trusted user not being very active in the trust network (trusts 45 people compared to the 1819 trusts of the most trusting user), does not seem to care as much about the preference models of others in the network and the majority of his followers and followees are assigned to row clusters different from him i.e. row cluster 4 and row cluster 1. Interestingly the majority of the followers of the most trusted user are also likely the most trusting users in the trust network and it is no surprise that both are assigned the same row cluster (i.e. row cluster 4).

Both spectral dimensionality reduction and sparse SVD were successful in capturing the most variation in terms of the eigenvectors in a small number of dimensions (i.e. 10 compared to the 120491 users in Epinions and 147610

users in Flixster). Since sparse SVD was used to reduce the dimensions of both the user trust network and the item network, we ended up with dense features. Thus we did not use a regularization such as lasso which would enforce sparsity in the feature space. However, on a random 80/20 training/test split on a very small (485 users by 100 items) subset of Epinions, we represented subject category information for an item using 1-of-k representation. Using such sparse item features in addition to the user trust network, we found that using lasso (with $\lambda_{lasso} = 1.9$) gave a much lower MSE of 0.8061 compared to ridge (with $\lambda_{ridge} = 1.392$) which gave an MSE of 1.392. This was for 2 row clusters and 2 column clusters.

Approximately 37% of the 120491 users and 20% of the 755760 items in epinions are cold-start. On comparing the results for SCOAL and SS-SCOAL on validation set 1, SS-SCOAL seemed to perform worse with an MSE of 0.3442 compared to SCOAL's MSE of 0.2247. λ_R and λ_C are regularization terms that control the penalty for the user and item feature being very different from the co-cluster mean user and item feature. We only tried specific values for λ_R (=100) and λ_C (=100) and need to experiment further before drawing any conclusions.

Table 3.9 shows the RMSE results for different approaches evaluated by Jamali et al. [11]. The best RMSE we got on the Flixster dataset is 0.916 with a standard deviation of 0.0014 using (K,L)=(6,6) and $\lambda_{ridge} = (6, 6)$. This makes our approach comparable to collaborative filtering but worse than the MF (matrix factorization) approaches. MF approaches however lack

Approach	K=5	K=10
CF	0.911	0.911
Base MF	0.878	0.863
STE	0.864	0.852
Social MF	0.821	0.815

Table 3.9: RMSE values for comparison partners on Flixster with varying dimensionality of latent factors (K). Table is taken from Social MF [11]

Data Statistics	Data Set used in Social MF [11]	Data Set made publicly available by [11]
Users	1M	1.049508M
Social Relations	26.7M	7.058819M
Ratings	8.2M	8.196077M
Items	49K	48.794K
Users with Rating	150K	147.612K
Users with Friend	980K	186.564K

Table 3.10: Data Statistics comparing the data set collected in [11] and the one that was made publicly available by the same authors.

interpretability because it is not clear what the latent features represent. It is also not clear which latent features are the most influential. In SCOAL, the feature weights across different co-clusters could be compared to see which features were the most weighted, different and thus most influential in affecting a recommendation. One could also recommend users in the same co-cluster as possible people to trust since they have been found to have similar preference models. The low dimensional user, item features within a co-cluster could be compared to see how much of the co-clustering is affected by similarity in terms of the trust or item information. Users are more likely to trust a recommendation if they know the reasoning behind the recommendation. The Flixster dataset did not contain additional side information such as movie genres, etc which SCOAL could have benefited from. MF approaches would not be able to handle such side information.

Jamali mentions on his website that the Flixster "data set is a cleaned version of the data set we used for RecSys 2010, and not the exact one." We have requested more information from Jamali et al. about the kind of cleaning they did but have yet to hear back from them. As seen in Table 3.10, the datasets seem to be the same except that there are fewer social relations in the publicly available dataset. Using more social network information, on par with the MF approaches could perhaps have given better results.

Chapter 4

Label Propagation

4.1 MAD

Label propagation algorithms ([29],[2]) spread label distributions from a small set of labeled nodes with some initial label information, throughout the graph. Modified Adsorption (MAD) [27] is a modification of the Adsorption algorithm [2] which transductively propagates labels from the labeled nodes to the unlabeled nodes in a graph. Talukdar et al.[27] prove that Adsorption does not have an objective function and propose MAD as a modification of Adsorption with a well defined objective function.

The spreading of the label distributions can be viewed as a controlled random walk with three possible actions: (i) injecting a seeded node with its seed label, (ii) continuing the walk from the current node to a neighboring node, and (iii) abandoning the walk. MAD takes three parameters, μ_1, μ_2 and μ_3, which control the relative importance of each of these actions, respectively. The objective function for MAD is as follows:

$$C_{mad}(\hat{Y}) = \sum_{l}[\mu_1(Y_l - \hat{Y}_l)^T S(Y_l - \hat{Y}_l) + \mu_2 \hat{Y}_l^T L \hat{Y}_l + \mu_3 ||\hat{Y}_l - R_l||_2^2] \quad (4.1)$$

where $l \in \{1, ..., m\}$ are the labels, $Y \in R_{+}^{n*m}$ is a matrix whose rows correspond to the nodes in the graph and columns correspond to the labels. Each row represents the labeling distribution for a node. For a node v in the graph, Y_v encodes the prior knowledge while $\hat{Y}_v$ is the predicted labeling distribution. The terms S, L and R are functions of the probability of injecting (ρ_{inj}), probability of continuing (ρ_{cont}) and the probability of abandoning the random walk (ρ_{abnd}) respectively.

The three terms in the objective function for MAD capture three requirements. First that the output of the algorithm be as close as possible to the a-priori labels for the labeled nodes. Second that the labeling respect the graph structure i.e. nodes close to each other in the graph, have similar labels. Third that the output be as un-informative as possible.

An extension of MAD, named MADDL [27] is particularly suited for ordinal labels. The objective function for MADDL is the same as MAD but for an additional term that penalizes the algorithm if similar labels are assigned different scores. The parameter μ_4 controls the penalty for this additional term. The objective function for MADDL is defined as follows

$$C_{maddl}(\hat{Y}) = C_{mad}(\hat{Y}) + \mu_4 * \sum_{i} \sum_{l,l'} C_{l,l'}(\hat{Y_{il}} - \hat{Y_{il'}})^2 \tag{4.2}$$

where each entry $C_{ll'}$ of the matrix C represents the dependence or similarity between the labels l and l'.

We use MAD over a graph with nodes representing user-item pairs,

individual users and items. The rating values (e.g. 1 through 5) representing the labels to be propagated. An edge connects each user-item pair (u, i) to the constituent user u and item i. An edge between user u and user v indicates that the users u and v are connected in the trust network. An edge between items i and j indicates that items i and j share the same author or subject. This allows the user-item pair (u, i) to be indirectly connected to user-item rating pair (v, j) if user u and user v are friends in the trust network or if items i and j share the same subject and/or the same author. Capturing the connections of user-item pairs in this indirect manner has the advantage of lowering the algorithm complexity both in terms of time and memory (from $O(N^2)$ to $O(N)$ where N is the number of ratings). Since MADDL takes in a similarity matrix for the labels, we defined the similarity between labels l and l' as $1/abs(l - l')$. All edge weights are kept as ones.

MADDL has been successfully used for predicting the ratings of electronic product reviews. "Each review is assigned one of four scores: 1 (worst), 2, 3, 4 (best). The reviewer name and location, a product name, a review title and date, and the review text is available. A K-NN graph is created from the reviews by using cosine similarity as the measure of similarity between reviews. Five training-test splits were created and precision was used as the evaluation metric".[27] Since we did not have the actual review text for the reviews on Epinions we did not build such a K-NN graph.

In our previous work with Speriousu et al.[25], MAD was used to classify sentiment of tweets. We propagated sentiment labels from a maximum

entropy classifier trained on noisy labels and knowledge about word types encoded in a lexicon, in combination with the Twitter follower graph. Results on polarity classification for several Twitter datasets showed that the label propagation approach rivals a model supervised with in-domain annotated tweets and outperforms the noisily supervised classifier it exploits as well as a lexicon-based polarity ratio classifier. Thus MAD has been shown to be successful in capturing side information such as lexicon information to predict sentiment. Sentiment here can be likened to a recommendation.

Jamali et al.[10] proposed a random walk method, TrustWalker that combines trust-based and item-based recommendation. Each random walk on the user trust graph returns a predicted rating for user u on target item i. The probability of stopping is directly proportional to the similarity between the target item and the most similar item j, weighted by the sigmoid function of step size k. The more the similarity, the greater the probability of stopping and using the rating on item j as the predicted rating for item i. As the step size increases, the probability of stopping decreases. Thus ratings by closer friends on similar items are considered more reliable than ratings on the target item by friends further away.

The random walk of TrustWalker differs from the one in MAD since the probability of jumping from user u to user v in the graph is based on the degree and not the similarity between item i and the items that v has rated. Moreover the graph in TrustWalker is the user-user trust graph. MAD can also have items and item features as nodes in the graph. A limitation of TrustWalker is

Input:
- **Graph:** $G = (V, E, W)$
- **Prior labeling:** $\mathbf{Y}_v \in \mathbb{R}^{m+1}$ for $v \in V$
- **Probabilities:** $p_v^{inj}, p_v^{cont}, p_v^{abnd}$ for $v \in V$

Output:
- **Label Scores:** $\hat{\mathbf{Y}}_v$ for $v \in V$

1: $\hat{\mathbf{Y}}_v \leftarrow \mathbf{Y}_v$ for $v \in V$ {Initialization}
2: $\mathbf{M}_{vv} \leftarrow \mu_1 \times p_v^{inj} + \mu_2 \sum_{u \neq v} (p_v^{cont} \mathbf{W}_{vu} + p_u^{cont} \mathbf{W}_{uv}) + \mu_3$
3: **repeat**
4: $\quad D_v \leftarrow \sum_u \left(p_v^{cont} \mathbf{W}_{vu} + p_u^{cont} \mathbf{W}_{uv}\right) \hat{\mathbf{Y}}_u$
5: $\quad$ **for all** $v \in V$ **do**
6: $\quad\quad \hat{\mathbf{Y}}_v \leftarrow \frac{1}{\mathbf{M}_{vv}} \left(\mu_1 \times p_v^{inj} \times \mathbf{Y}_v + \mu_2 \times D_v + \mu_3 \times p_v^{abnd} \times \mathbf{r}\right)$
7: $\quad$ **end for**
8: **until** convergence

Figure 4.1: MAD algorithm

that unlike SCOAL and MAD, it cannot use available side information for the users, items. Such side information (for e.g. user demographic information, item subject categories etc) can be valuable in predicting a user's rating on an item.

4.2 Results

Since MAD is set in a transductive setting, the ratings in the training set are equivalent to the seeded nodes and the algorithm is evaluated based on the MSE computed on the unlabeled rating nodes i.e the ratings in the test set.

Figure 4.1 details the algorithm for MAD. In every iteration, the predicted labels ($\hat{Y}_l$) are updated independent of each other. This makes the

MAD algorithm memoryless and thus easily parallelizable. We used Junto [1], an implementation of MAD in Java. Although MAD is very scalable, its current implementation is not. The current implementation ran out of memory once the number of edges exceeded 77,00,000 on a 64-bit, 24 core machine, allocating 16GB to the JVM. Since the Flixster, Epinions datasets were an order of magnitude larger, we limited our analysis to a subset of 978 users and 1000 items from the Epinions dataset. A Hadoop based implementation of MAD is now available in Junto, though it came too late to be used in this thesis.

Table 4.1 shows the results for MAD on sorting the ratings by time and using the first 80% as training or labeled nodes and the rest as test or unlabeled nodes. Table 4.1 only uses the trust relationships in the user trust network.

Tables 4.2, 4.3 and 4.4 show the results from cross-validation on MAD when (i) using user and item information i.e. user-item rating pairs are indirectly connected if they rated the same item or were rated by the same user (ii) including trust, distrust information and (ii) including trust, distrust network along with subject and author information for items. Table 4.5 shows the same for SCOAL with the low dimensional embeddings of the user trust-distrust network and the item-item network.

Table 4.6 shows the MSE using MADDL for varying values of μ_1, μ_2,

[1] http://code.google.com/p/junto

Number of iterations	μ_1	μ_2	μ_3	Average MSE
100	0.005	0.01	0.01	1.5318
200	0.005	0.01	0.01	1.5318
500	0.005	0.01	0.01	1.5318
1000	0.005	0.01	0.01	1.5318
200	**1**	**0.5**	**1**	**1.5318**
100	0.0001	0.01	0.01	1.5318
100	0.001	0.01	0.01	1.5318
100	100	0.01	0.01	1.5318
100	0.01	0.1	0.01	1.5058
100	0.01	0.01	0.1	1.5318

Table 4.1: MAD on the 978 users by 1000 items subset from Epinions using only the trust relationships in the user trust network. In bold are the results for the Adsorption algorithm (which turns out to be a special case of MAD) that was used as a baseline in [27].

μ_3 and μ_4.

4.3 Discussion

As seen in Table 4.1, MAD beat Adsorption, which is the baseline in the original paper on MAD [27]. Using directed edges for the users i.e. an edge from user v to u means that user u trusts user v, did not lead to any change in the results. Introducing item subject and author information reduced the MSE from 1.138 to 1.114 (see tables 4.3 and 4.4). The lowest MSE was for ($\mu_1 = 1$, $\mu_2 = 0.01$, $\mu_3 = 0.01$). Thus a larger weight for injecting the node with its seed label was preferred.

SCOAL did significantly better than MAD with a lowest MSE of 0.7247

μ_1	μ_2	μ_3	Average MSE from five fold cross validation
1	0.01	0.01	**1.1098** ($\pm$0.0162)
0.01	1	0.01	1.2301 ($\pm$0.0147)
1	1	0.01	1.2299 ($\pm$0.0149)
1	1	1	1.1879 ($\pm$0.0472)
100	100	1	1.2299 ($\pm$0.0149)
0.2	0.7	0.1	1.2302 ($\pm$0.0147)
0.7	0.2	0.1	1.1574 ($\pm$0.0143)
0.2	0.1	0.7	1.1833 ($\pm$0.0676)

Table 4.2: MAD on the 978 users by 1000 items subset from Epinions. The graph consists of edges between rating pairs and users, items.

μ_1	μ_2	μ_3	Average MSE from five fold cross validation
1	0.01	0.01	**1.138** ($\pm$0.0127)
0.01	1	0.01	1.1394 ($\pm$0.0115)
1	1	0.01	1.14 ($\pm$0.0136)
0.01	0.01	1	1.1388 ($\pm$0.0128)
1	1	1	1.1406 ($\pm$0.0122)
100	100	1	1.1388 ($\pm$0.0133)

Table 4.3: MAD on the 978 users by 1000 items subset from Epinions. The graph includes user trust and distrust relationships in addition to edges between rating pairs and users, items.

μ_1	μ_2	μ_3	Average MSE from five fold cross-validation	Average MSE from five fold cross-validation on 20/80 split
1	0.01	0.01	**1.114** (±0.0148)	**1.1607** (±0.0056)
0.01	1	0.01	1.2298 (±0.015)	1.2301 (±0.0037)
1	1	0.01	1.2298 (±0.015)	1.2301 (±0.0036)
1	1	1	1.1648 (±0.014)	1.1749 (±0.0072)
100	100	1	1.2298 (±0.015)	1.2301 (±0.0037)
0.2	0.7	0.1	1.2302 (±0.0147)	1.2301 (±0.0037)
0.7	0.2	0.1	1.1624 (±(0.0143)	1.1980 (±0.0106)
0.2	0.1	0.7	1.1124 (±0.0167)	1.1929 (±0.051)

Table 4.4: MAD on the 978 users by 1000 items subset from Epinions when using trust and distrust, item-subject and item-author information.

Features	Global baseline (1,1)	(2,2)	(3,3)	(4,4)	(5,5)
Trust and Distrust	0.7769 (±0.0066)	**0.722** (±0.0058)	0.7265 (±0.005)	0.7302 (±0.0042)	0.736 (±0.0062)
Trust, Distrust, Item-Subject-Author information	0.7774 (±0.0062)	**0.7243** (±0.0085)	0.7263 (±0.0048)	0.7301 (±0.0055)	1.0415 (±0.6806)
Trust and Distrust with 20/80 train/test split	**0.7786** (±0.0026)	0.8122 (±0.0103)	0.8462 (±0.0126)	0.8640 (±0.0141)	0.8726 (±0.0107)

Table 4.5: Mean Squared Error for varying (k, l) where k is the row cluster size and l is the column cluster and $\lambda_{ridge} = 10^4$. MSE averaged over 4 runs for five fold cross validation on the 978 users by 1000 items subset . The best results are in bold.

μ_1	μ_2	μ_3	μ_4	Average MSE from five fold cross-validation
1	0.01	0.1	0.01	**1.115** ($\pm$0.0171)
1	0.01	0.01	0.01	1.127 ($\pm$0.0178)
0.25	0.25	0.25	0.25	1.1672 ($\pm$0.0152)
0.50	0.125	0.125	0.125	1.1604 ($\pm$0.0150)
200	100	100	100	1.1648 ($\pm$0.0148)
0.125	0.50	0.125	0.125	1.182 ($\pm$0.0159)
0.125	0.125	0.50	0.125	1.14 ($\pm$0.017)
0.125	0.125	0.125	0.50	1.1842 ($\pm$0.015)

Table 4.6: MADDL on the 978 users by 1000 items subset from Epinions when using trust, distrust, item-subject and item-author information.

as compared to MAD's MSE of 1.109. Since label propagation algorithms were designed with the goal of least supervision (i.e. fewer labeled nodes) in mind, we compared MAD and SCOAL on a 20/80 labeled/unlabeled split. To our surpise, we found that SCOAL continued to do better than MAD. As a sanity check, both algorithms individually did worse on the 20/80 split as compared to the 80/20 labeled/unlabeled split.

The significantly high error of MAD, even compared to predicting the mean (0.8220) indicates that the probabilities seem to be drifting away from the seeded values. This might be due to a large number of un-seeded nodes coming in between the seeded nodes. We used a very simple weighting scheme of weighing the edges as 1s for trust and -1s for distrust. Since the connections between the edges are supposed to make them similar, modeling distrust in this manner might be incorrect. A lot more could be done with respect to choosing different graph structures and weighing the edges.

MADDL seemed well suited to the rating prediction task since our labels or rating values are ordinal. A rating of 5 would be closer to a rating of 4 as compared to a rating of 1. It would be beneficial to have the additional constraint that the final labeling of the nodes respect such ordering i.e. the labeling distribution for a node with initial rating 4 be similar to the labeling distribution for a node with initial rating 5. Empirically it showed almost equivalent results with an MSE 1.115 (see Table 4.6). This lowest MSE was for ($\mu_1 = 1$, $\mu_2 = 0.01$, $\mu_3 = 0.1$, $\mu_4 = 0.01$).

Surprisingly, using just user, item and rating information gave an MSE

in the same range as when including trust/distrust, item-subject, item-author information. This seems to imply that the trust information is not of much use when using a framework like MAD. This is to be taken with a pinch of salt since we only evaluated on a small subset. Scaling the current implementation of MAD and using MAD on the entire dataset might suggest otherwise.

The probability distribution over all possible rating values in MAD could be used as a confidence measure for the predicted rating. Significance tests on the feature weights for the co-clusters in SCOAL could also be used as confidence measures. A by-product of MAD is a rating distribution over individual users and items. This could be used to rank users and items based on who are more likely to give or get high ratings. A user who has a high probability of rating 5 could be recommended to rate more product reviews to improve the marketability of the product. The review that has a high probability of being rated 5 could be seen as valuable and given a greater rank in search results.

Chapter 5

Conclusions and Future Work

Recommender systems can benefit from combining user-item rating history with information from the social/trust network of a user. We explore two ways of doing this: (1) simultaneous co-clustering and learning (SCOAL) [5] of the user-item rating matrix with a low rank embedding of the social network and (2) modified adsorption (MAD) [27] which propagates ratings of user-item pairs directly on the user social graph. We evaluate both approaches on two large public data-sets from Epinions.com and Flixster.com.

Related work in the field has primarily focussed on trust propagation([7], [20], [10]) and MF based approaches([11]). Since distrust is not as transitive as trust i.e. an enemy's enemy need not be an enemy or a friend, distrust can't directly replace trust in trust propagation approaches. By using a low dimensional representation of the original trust network in SCOAL, we use distrust as it is and don't propagate it. We find that using trust information and in particular distrust information results in better prediction of ratings.

The Matrix Factorization (MF) based approaches lack interpretability since it is not clear what the latent features represent and it is hard to reason

about the rationale of a user. Both SCOAL and MAD are highly interpretable and allow us to reason about the user's prediction on a rating.

SCOAL captures the preference models for certain users in the trust network for certain groups of items. Results indicate that local preference models on the entire Epinions and Flixster datasets, outperform the global preference model. Using SCOAL, we can pin-point the groups of users and the groups of items that have the same preference model. The co-clusters in SCOAL can be likened to communities of users in a trust network sharing their tastes for particular types of movies or reviews. The feature weights across different co-clusters can be compared to see which features are the most weighted, different and thus most influential in affecting a recommendation.

Both SCOAL and MAD are able to seamlessly integrate side information such as item-subject and item-author information into the trust based rating prediction model. A by-product of MAD is a rating distribution over individual users and items. This could be used to rank users and items based on who are more likely to give or get high ratings. Such information is particularly useful for marketing and ranking prediction problems.

MAD seemed to do poorly on a small subset of the Epinions data. However scaling the current implementation of MAD and using MAD on the entire dataset might suggest otherwise. A Hadoop based implementation of MAD was recently released, though it came too late to be used in this thesis.

We used a very simple weighting scheme of weighing the edges as 1s

for trust and -1s for distrust. Since the connections between the edges are supposed to make them similar, modeling distrust in this manner might be incorrect. A lot more could be done with respect to choosing different graph structures and weighing the edges. An alternative to the item-item network would be a graph where in the items, subjects and authors themselves serve as nodes and the edges capture the relationships between the nodes.

In the future, we intend to use different input graph configurations and see if this leads to improvement for MAD. Since Epinions also has timing information for ratings, we could use time based cell level features in SCOAL to improve on the rating prediction. We also seek to do a more extensive evaluation and compare with other social rating network approaches on common datasets.

Appendices

Appendix A

Snippets of the Datasets

A.0.1 Extended Epinions dataset [19]

The dataset consists of three files—ratings.txt, mc.txt (which contains details about the review article) and user_rating.txt (which contains user trust information). Tables , and list the first five lines from each of these files.

Each article is written by a single user. The header information for ratings.txt is as follows:

- CONTENT_ID: The object ID of the review article.
- AUTHOR_ID: The ID of the user who wrote the article.
- SUBJECT_ID: The ID of the subject that the article is about.

The header information for mc.txt is as follows:

CONTENT-ID	AUTHOR-ID	SUBJECT-ID
1445594	718357	149002425217
1445595	220568	149003604865
1445596	717325	5303145344
1445597	360156	192620893057
1445598	718857	149002163073

Table A.1: First five lines of the Epinions article information file (mc.txt)

1	2	3	4	5	6	7	8
891503	311238	5	0	2001/01/10	2001/12/27	1	2522499
891503	210412	5	0	2001/01/10	2001/12/27	1	2522499
891503	351471	5	0	2001/01/10	2001/12/27	1	2522499
891503	394639	5	0	2001/01/10	2001/12/27	1	2522499
891503	200338	4	0	2001/01/10	2001/12/27	1	

Table A.2: Last five lines of the Epinions ratings file (ratings.txt)

1. OBJECT_ID: The object ID of the object being rated.

2. MEMBER_ID: The ID of the member who rated the object.

3. RATING: The rating on a scale from 1-5 (1- Not helpful , 2 - Somewhat Helpful, 3 - Helpful 4 - Very Helpful 5- Most Helpful). We followed the author's recommendation of treating the 6s as 5s.

4. STATUS: The display status of the rating. 1 indicates that the member has chosen not to show his rating of the object and 0 indicates that the member does not mind showing his name beside the rating.

5. CREATION: The date on which the member first rated the object.

6. LAST_MODIFIED: The last date on which the member modified his rating of the object.

7. TYPE: If and when Epinions allows more than just content rating to be stored in this table, this column would store the type of the object being rated.

8. VERTICAL_ID: Vertical_id of the review article.

MY-ID	OTHER-ID	VALUE	CREATION
3287060356	232085	-1	2001/01/10
3288305540	709420	1	2001/01/10
3290337156	204418	-1	2001/01/10
3294138244	269243	-1	2001/01/10

Table A.3: First five lines of the Epinions trust file (user_rating.txt)

user-id	movie-id	rating
882359	81	1.5
882359	926	1
882359	1349	2
882359	2270	1
882359	3065	5

Table A.4: First five lines of the Flixster ratings file

The header information for the user_rating.txt file is as follows:

- MY_ID: ID of the member making the trust/distrust statement.
- OTHER_ID: The ID of the member being trusted/distrusted.
- VALUE: Value = 1 for trust and -1 for distrust.
- CREATION: The date on which a trust was created.

A.0.2 Flixster dataset [11]

Tables and list the first five lines from the ratings and trust information files for Flixster. The headers for these files are self-explanatory.

user-id	friend-id
6	1761
6	1770
6	1775
6	24
6	1785

Table A.5: First five lines of the Flixster friends file

Appendix B

SCOAL Code Vectorization

As seen from the Profiler snapshot in Figure B.1, the primary bottleneck in performance was in PredictMissingF.m which uses the model learnt i.e. the feature weights (betacoeff), the row cluster and column cluster assignments to predict the rating (as $\hat{r} = X^*$[betacoeff]). We found that the bottleneck was primarily due to the presence of an expensive "for loop" which looped as many times as the number of data points we predict. The for loop was thus replaced by vectorized code which lead to significant speedup.

The details of the vectorization are as follows. Previously, we were looping around the number of entries to predict and concatenating their feature coefficients together. Instead, we now stack up the assigned row and column cluster numbers as many times as the number of features and then directly index into the betas matrix to create the concatenated list of feature coefficients.

Preliminaries:

- m_rows are the row indices of the entries to predict
- m_cols are the col indices of the entries to predict

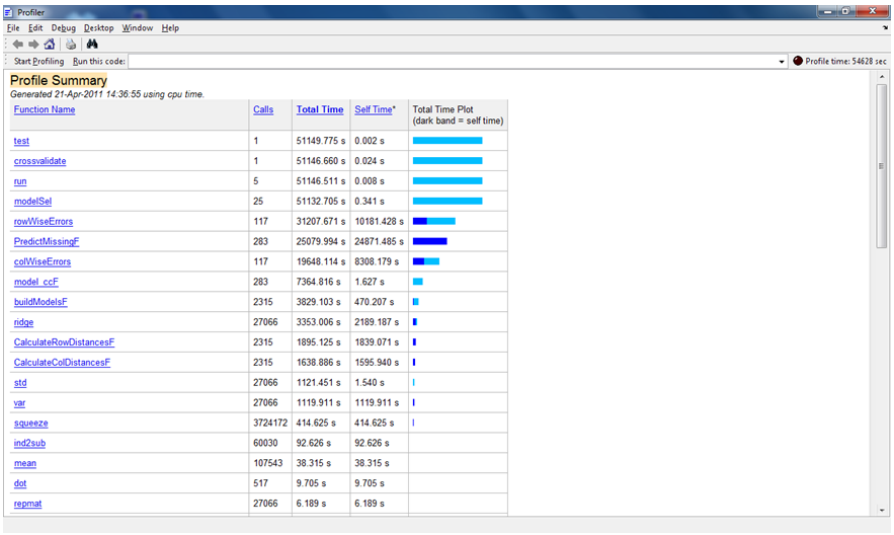

Figure B.1: Profiler Snapshot before vectorization.

- betas is a (number of row clusters) x (number of col clusters) x (number of features) matrix containing the learnt feature weights.
- rowClust contains the assigned row cluster numbers for the entries to predict
- colClust contains the assigned col cluster numbers for the entries to predict

Original code:

```
beta_coeff = [];
for ii = 1:size(m_rows, 1)
    beta_coeff = [beta_coeff ;
    squeeze(betas(rowClust(ii), colClust(ii), :))'];
end
```

Vectorized code:

```
beta_coeff = [];
numfeatures = size(X, 2);
B1 = repmat(rowClust, numfeatures, 1);
B2 = repmat(colClust, numfeatures, 1);
B3 = reshape(repmat(1:numfeatures, length(m_vals), 1), 1, [])';
B = sub2ind(size(betas), B1, B2, B3);
B = reshape(B, length(m_vals), numfeatures);
fakeBetas = betas(:);
beta_coeff = fakeBetas(B);
```

The above code and the cross-validation code for splitting, testing is available at /home/nsudan/Thesis/ on the IDEAL (Intelligent Data Exploration and Learning) lab machines. The same directory also contains a version of SCOAL that only uses user features. We are currently in the process of implementing cell level features for SCOAL. Neither of the two datasets used in this thesis used cell level features.

Bibliography

[1] Meghana Deodhar Aayush Sharma and Joydeep Ghosh. Simultaneous co-clustering and learning for large scale recommendations. Technical report, The University of Texas at Austin, 2009.

[2] Shumeet Baluja, Rohan Seth, D. Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. Video suggestion and discovery for youtube: taking random walks through the view graph. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 895–904, New York, NY, USA, 2008. ACM.

[3] Yoshua Bengio, Olivier Delalleau, Nicolas Le Roux, Jean-Francois Paiement, Pascal Vincent, and Marie Ouimet. Spectral dimensionality reduction. Technical report, CIRANO, 2004.

[4] Yizong Cheng and George M. Church. Biclustering of expression data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103. AAAI Press, 2000.

[5] Meghana Deodhar and Joydeep Ghosh. A framework for simultaneous co-clustering and learning from complex data. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 250–259, New York, NY, USA, 2007. ACM.

[6] Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 89–98, New York, NY, USA, 2003. ACM.

[7] Jennifer Ann Golbeck. *Computing and applying trust in web-based social networks*. PhD thesis, College Park, MD, USA, 2005. AAI3178583.

[8] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35:61–70, 1992.

[9] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 230–237, 1999.

[10] 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*, KDD '09, pages 397–406, New York, NY, USA, 2009. ACM.

[11] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceed-*

ings of the fourth ACM conference on Recommender systems, RecSys '10, pages 135–142, New York, NY, USA, 2010. ACM.

[12] Mohsen Jamali, Gholamreza Haffari, and Martin Ester. Modeling the temporal dynamics of social rating networks using bidirectional effects of social relations and rating patterns. In *Proceedings of the 2010 IEEE International Conference on Data Mining Workshops*, ICDMW '10, pages 344–351, Washington, DC, USA, 2010. IEEE Computer Society.

[13] George Karypis, Rajat Aggarwal, Vipin Kumar, and Shashi Shekhar. Multilevel hypergraph partitioning: application in vlsi domain. In *Proceedings of the 34th annual Design Automation Conference*, DAC '97, pages 526–529, New York, NY, USA, 1997. ACM.

[14] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[15] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, CIKM '03, pages 556–559, New York, NY, USA, 2003. ACM.

[16] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, December 2007.

[17] 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, SIGIR '09, pages 203–210, New York, NY, USA, 2009. ACM.

[18] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 931–940, New York, NY, USA, 2008. ACM.

[19] Paolo Massa and Paolo Avesani. Controversial users demand local trust metrics: an experimental study on epinions.com community. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 1*, pages 121–126. AAAI Press, 2005.

[20] Paolo Massa and Paolo Avesani. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, pages 17–24, New York, NY, USA, 2007. ACM.

[21] Yutaka Matsuo and Hikaru Yamamoto. Community gravity: measuring bidirectional effects by trust and rating on online social networks. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 751–760, New York, NY, USA, 2009. ACM.

[22] Viet-An Nguyen, Ee-Peng Lim, Jing Jiang, and Aixin Sun. To trust or not to trust? predicting online trusts using trust antecedent framework. In *Data Mining, 2009. ICDM '09. Ninth IEEE International Conference on*, pages 896 –901, dec. 2009.

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

[24] Rashmi R. Sinha and Kirsten Swearingen. Comparing recommendations made by online systems and friends. In *DELOS Workshops*, 2001.

[25] Michael Speriosu, Nikita Sudan, Sid Upadhyay, and Jason Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the EMNLP 2011 Workshop on Unsupervised Learning*, 2011.

[26] Panagiotis Symeonidis, Eleftherios Tiakas, and Yannis Manolopoulos. Transitive node similarity for link prediction in social networks with positive and negative links. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 183–190, New York, NY, USA, 2010. ACM.

[27] Partha Talukdar and Koby Crammer. New regularized algorithms for transductive learning. In Wray Buntine, Marko Grobelnik, Dunja Mladenic, and John Shawe-Taylor, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5782, pages 442–457. Springer Berlin / Heidelberg, 2009.

[28] Chun-Yuen Teng, Debra Lauterbach, and Lada A. Adamic. I rate you. you rate me. should we do so publicly? In *Proceedings of the 3rd conference on Online social networks*, WOSN'10, pages 12–12, Berkeley, CA, USA, 2010. USENIX Association.

[29] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.