New Algorithms for Social Recommendation

Joseph Christian G. Noel

A subthesis submitted in partial fulfillment of the degree of
Master of Computing (Honours) at
The Department of Computer Science
Australian National University

October 2011

Contents

Acknowledgements **vii**

Abstract **ix**

1 Introduction **1**
1.1 Objectives 1
1.2 Contributions 3
1.3 Outline 4

2 Background **7**
2.1 Notation 7
2.2 Related work 8
2.2.1 Collaborative Filtering Algorithms 8
2.2.2 Matrix Factorization Models 9
2.2.3 Social Collaborative Filtering 10
2.2.4 Tensor Factorization Methods 10

3 Evaluation of Social Recommendation Systems **13**
3.1 Facebook 13
3.1.1 LinkR 13
3.2 Dataset 14
3.2.1 User Data 14
3.2.2 Link Data 15
3.2.3 Implicit Dislikes 15
3.3 Evaluation Metrics 15
3.4 Training and Testing Issues 16
3.4.1 Training Data 16
3.4.2 Live Online Recommendations 17
3.4.3 Test Data 17

4 Comparison of Existing Recommender Systems **19**
4.1 Objective components 19
4.1.1 Matchbox Matrix Factorization 19
4.1.2 L2 Regularization 19
4.1.3 Social Regularization 20
4.1.4 Derivatives 20
4.2 Algorithms 22
4.3 Online Results 22

4.4 Offline Results 23
4.5 Survey Results 24

5 New Algorithms for Social Recommendation **31**
5.1 New Objective Components 31
5.1.1 Hybrid Objective 31
5.1.2 Social Spectral Regularization 32
5.1.3 Social Co-preference Regularization 32
5.1.4 Social Co-preference Spectral Regularization 33
5.1.5 Derivatives 33
5.2 Second Trial 36
5.2.1 Online Results 36
5.3 Offline Results 37

6 Conclusion **43**

Bibliography **45**

Chapter 1

Introduction

Given the vast amount of content available on the Internet, finding information of personal interest (news, blogs, videos, movies, books, etc.) is often like finding a needle in a haystack. Recommender systems based on *collaborative filtering* (CF) aim to address this problem by leveraging the preferences of a user population under the assumption that similar users will have similar preferences. These principles underlie the recommendation algorithms powering websites like Amazon and Netflix.[1]

As the web has become more social with the emergence of Facebook, Twitter, LinkedIn, and most recently Google+, this adds myriad new dimensions to the recommendation problem by making available a rich labeled graph structure of social content from which user preferences can be learned and new recommendations can be made. In this socially connected setting, no longer are web users simply described by an IP address (with perhaps associated geographical information and browsing history), but rather they are described by a rich user profile (age, gender, location, educational and work history, preferences, etc.) and a rich history of user interactions with their friends (direction comments/posts, clicks of like, tagging in photos, mutual group memberships, etc.). This rich information poses both an amazing opportunity and a daunting challenge for machine learning methods applied to social recommendation — how do we fully exploit the social network content in recommendation algorithms?

1.1 Objectives

This paper examines the problem of designing efficient, scalable, and accurate *social CF* (SCF) algorithms for *personalized link recommendation on Facebook* – quite simply the task of recommending personalized links to users that might interest them. *User interest* can be determined via many methods including *indirect feedback* in the form of link clicks and *direct feedback* in the form of explicit link ratings or other evidence that a user *liked* a link (e.g., explicitly clicking "like").

Many existing SCF approaches extend *matrix factorization* (MF) techniques for CF [Salakhutdinov and Mnih 2008] and have proved quite powerful in their ability to

[1] On Amazon, this is directly evident with statements displayed of the form "users who looked at item X ended up purchasing item Y 90% of the time". While the exact inner workings of Netflix are not published, the best performing recommendation algorithm in the popular Netflix prize competition [Toscher and Jahrer 2009] used an ensemble of CF methods.

accurately model user preferences even when only a unique ID is available for both the user and item being recommended. The power of such methods stems from their ability to project users and items into latent vector spaces of reduced dimensionality where they are effectively grouped by similarity. Indeed, we will show in Chapter 4 that existing social extensions of MF are quite powerful and outperform a variety of other commonly used SCF approaches.

Given the strong performance of existing MF approaches to SCF, we aim to comparatively evaluate them and further improve on their performance where possible. To do this, we first identify a number of problems of existing SCF MF methods that we make our objective to address in this paper:

(a) **Non-feature-based user similarity:** Existing SCF MF methods do not permit the use of item or link features in learning user similarity based on observed interactions.

(b) **Model direct user-user information diffusion:** Existing SCF MF methods do not permit directly modeling user-user information diffusion according to the social graph structure.

(c) **Restricted common interests:** Existing SCF MF methods cannot learn that that two users may only have overlapping interests in specific areas.

This paper addresses all of these problems with novel contributions in an efficient, scalable, and unified latent factorization component framework for SCF. We present results of our algorithms on live trials in a custom-developed Facebook App involving data collected over three months from over 100 App users and their nearly 30,000 friends. These results show that a number of extensions proposed to resolve (a)–(c) outperform all previously existing algorithms.

In addition, given that live online user evaluation trials are time-consuming, requiring many users and often an evaluation period of at least one month, we have one last important objective to address in this paper:

(d) **Idenfying passive evaluation paradigms that correlate with actively elicited human judgments.** The benefits of doing this are many-fold. When designing new SCF algorithms, there are myriad design choices to be made, for which actual performance evaluation is the only way to validate the correct choice. Furthermore, simple parameter tuning is crucial for best performance and SCF algorithms are often highly sensitive to well-tuned parameters. Thus for the purpose of algorithm design and tuning, it is crucial to have methods and metrics that can be evaluated immediately on passive data (i.e., a passive data set of user likes) that are shown to correlate with human judgments in order to avoid the time-consuming process of evaluating the algorithms in live human trials.

Next we outline our specific paper contributions to address the above problem objectives in-depth.

1.2 Contributions

In the preceding section, we outlined three deficiencies of existing MF approaches for SCF. Now we discuss our specific contributions in this paper to address these three deficiencies:

(a) **User-feature social regularization:** One can encode prior knowledge into the learning process using a technique known as *regularization*. In the case of social MF, we often want to regularize the learned latent representations of users to enforce that users who interact heavily often have similar preferences, and hence similar latent representations.

Thus to address the deficiency noted in *non-feature-based user similarity*, we build on ideas used in Matchbox [Stern et al. 2009] to incorporate user features into the social regularization objective for SCF. There are two commonly used methods for social regularization in SCF — in Chapter 5 we extend both to handle user features and determine that the *spectral* regularization extension performs best.

(b) **Hybrid social collaborative filtering:** While MF methods prove to be excellent at projecting user and items into latent spaces, they suffer from the caveat that they cannot model joint features over user and items (they can only work with independent user features and independent item features). This is problematic when it comes to the issue of *modeling direct user-user information diffusion* — in short, the task of learning how often information flows from one specific user to another specific user.

The remedy for this turns out to be quite simple — we need only introduce an objective component in addition to the standard MF objective that serves as a simple linear regressor for such information diffusion observations. Because the resulting objective is a combination of latent MF and linear regression objectives, we refer to it simply as *hybrid SCF*. In Chapter 5, we evaluate this approach and show that it outperforms standard SCF.

(c) **Copreference regularization:** Existing SCF methods that employ social regularization make a somewhat coarse assumption that if two users interact heavily (or even worse, are simply friends) that their latent representations must match as closely as possible. Considering that friends have different reasons for their friendships — co-worders, schoolmates, common hobby — it is reasonable to expect that two people (friends or not) may only share *restricted common interests*: co-workers may both enjoy technical content related to work, but differ otherwise; schoolmates may like to hear news about other schoolmates, but differ otherwise; people who share an interest in a common hobby are obviously interested in that hobby, but should not necessarily share common interests elsewhere.

To this end, we propose a finer-grained approach to regularizing users by restricting their latent user representation to be similar (or different) only in subspaces relevant to the items mutually liked/disliked (or disagreed upon – one user likes and the other dislikes). Because this method of regularization requires evidence

Contents

Chapter 2

Background

2.1 Notation

The following are mathematical notations common to the SCF setting and models explored in this paper:

- N users, each having an I-element feature vector $\mathbf{x} \in \mathbb{R}^I$ (alternately if a second user is needed, $\mathbf{z} \in \mathbb{R}^I$).
- M items, each having a J-element feature vector $\mathbf{y} \in \mathbb{R}^J$. The feature vectors for users and items can consist of any real-valued features as well as $\{0, 1\}$ features like user and item IDs.
- A (non-exhaustive) data set D of user preferences of the form $D = \{(\mathbf{x}, \mathbf{y}) \to R_{\mathbf{x},\mathbf{y}}\}$ where class $R_{\mathbf{x},\mathbf{y}} \in \{0 \text{ (dislike)}, 1 \text{ (like)}\}$.
- A (non-exhaustive) data set C of co-preferences derived from D of the form $C = \{(\mathbf{x}, \mathbf{z}, \mathbf{y}) \to P_{\mathbf{x},\mathbf{z},\mathbf{y}}\}$ where class $P_{\mathbf{x},\mathbf{z},\mathbf{y}} \in \{-1 \text{ (disagree)}, 1 \text{ (agree)}\}$.
- A similarity rating $S_{\mathbf{x},\mathbf{z}}$ between any users $\mathbf{x}$ and $\mathbf{z}$. This is used to summarize all social interaction between user $\mathbf{x}$ and user $\mathbf{z}$ in the term $S_{\mathbf{x},\mathbf{z}} \in \mathbb{R}$. A definition of $S_{\mathbf{x},\mathbf{z}} \in \mathbb{R}$ that has been useful is the following:

$$Int_{\mathbf{x},\mathbf{z}} = \frac{\#\text{ interactions between } \mathbf{x} \text{ and } \mathbf{z}}{\text{average } \#\text{ interactions between all user pairs}} \tag{2.1}$$

$$S_{\mathbf{x},\mathbf{z}} = \ln\left(Int_{\mathbf{x},\mathbf{z}}\right) \tag{2.2}$$

  For purposes of this definition, an *interaction* is any single event showing evidence that users $\mathbf{x}$ and $\mathbf{z}$ have interacted, e.g., a message exchange or being tagged in a photo together.

  In addition, we can define $S^+_{\mathbf{x},\mathbf{z}}$, a *non-negative* variant of $S_{\mathbf{x},\mathbf{z}}$:

$$S^+_{\mathbf{x},\mathbf{z}} = \ln\left(1 + Int_{\mathbf{x},\mathbf{z}}\right) \tag{2.3}$$

The matrix R is a sparse $N \times M$ matrix of user ratings on items. The problem of recommendation is filling out the empty elements of this matrix, and this can be looked at as a linear regression problem. There are two general ways that this has been done previously, Content-based Filtering (CBF) and Collaborative Filtering (CF). Content-based filtering makes recommendations based on correlations between the item features and the user's preferences on other items. In collaborative filtering, the system makes recommendations based on the correlation between other user's with similar preferences. Most traditional CBF methods learn in an explicit feature space, while most traditional CF methods learn in a latent feature space.

2.2 Related work

There is a massive amount of related work on SCF [Stern et al. 2009; Ma et al. 2009; Yang et al. 2011; Li et al. 2011; Rendle et al. 2009; Ma et al. 2008; Ma et al. 2011; Li and Yeung 2009; Cao et al. 2008; Cui et al. 2011] embodying some of the ideas above, however there are a few aspects covered here, not covered in this related work:

1. Existing SCF methods *cannot* capture some of the basic features that are used in standard CBF systems due to the inherent independent factorization between user and items (e.g., how much one user follows another) — this is the motivation behind the *hybrid* objectives.

2. All methods *except* for Matchbox [Stern et al. 2009] ignore the issue of user and item features. We extend the Matchbox approach above in our SCF methods.

3. *None* of the methods that propose social regularization [Ma et al. 2009; Ma et al. 2011; Li and Yeung 2009; Yang et al. 2011; Li et al. 2011; Cui et al. 2011] incorporate user features into this regularization (as done above).

4. Tensor-based factorizations such as [Rendle et al. 2009] use a full $K \times K \times K$ tensor for collaborative filtering w.r.t. tag prediction for users and items. While our co-preference regularization models above were motivated by tensor approaches, we instead take an item-reweighted approach to the standard inner products to (a) avoid introducing yet more parameters and (b) as a way to introduce additional regularization in a way that supports the standard Matchbox [Stern et al. 2009] CF model where prediction at run-time is made for a (user,item) pair, not for triples of (user,item,tag) as assumed in the tensor models.

2.2.1 Collaborative Filtering Algorithms

k-Nearest Neighbor

One of the most common form of CF is the nearest neighbor approach [Bell and Koren 2007]. The k-nearest neighbor algorithm is a method of pattern recognition that is based on the k closest training data in the feature space. There are two main variants of nearest neighbors for collaborative recommendation, user-based and item-based. Given a user u and an item i, let $N(u : i)$ be the set of nearest neighbors of u that have also given a rating for i, $N(i : u)$ be the set of nearest neighbors of i that have also been rated by u, $s_{uu'}$ the similarity rating between users u and u', and $s_{ii'}$ be the similarity rating for items i and i'. The predicted rating the user u gives item i in the user-based approach is then calculated as

$$r_{ui} = \frac{\sum_{v \in N(u;i)} s_{uv} r_{uv}}{\sum_{v \in N(u;i)} s_{uv}}$$

The item-based approach is calculated as

$$r_{ui} = \frac{\sum_{j \in N(i;u)} s_{ij} r_{uj}}{\sum_{j \in N(i;u)} s_{ij}}$$

The question of which approach to use depends on the dataset. When the number of items is far fewer than the number of users, it has been found that the item-based approach usually provides better predictions as well as being more efficient in computations.

Support Vector Machines

Support Vector Machines are a class of supervised learning classification algorithms that uses a hyperplane separating approach. During training, SVM builds a model by constructing a set of hyperplanes that separates one class of data from another class with the maximum margin possible. Data are classified by finding out on which side of a hyperplane they fall under.

For the experiments, SVM uses a fixed-length feature vector $\mathbf{f} \in \mathbb{R}^F$ derived from any $(\mathbf{x}, \mathbf{y}) \in D$, denoted as $\mathbf{f}_{\mathbf{x},\mathbf{y}}$. $\mathbf{f}_{\mathbf{x},\mathbf{y}}$ may include features that are non-zero only for specific items and/or users, e.g., a $\{0, 1\}$ indicator feature that user $\mathbf{x}$ and user $\mathbf{z}$ have both liked item $\mathbf{y}$. The implementation used for this paper is *LibSVM* [Chang and Lin 2001], which gives a regression score on the classification which can be used for ranking the results.

2.2.2 Matrix Factorization Models

As done in standard CF methods, we assume that a matrix U allows us to project users $\mathbf{x}$ (and $\mathbf{z}$) into a latent space of dimensionality K; likewise we assume that a matrix V allows us to project items $\mathbf{y}$ into a latent space also of dimensionality K. Formally we define U and V as follows:

$$U = \begin{bmatrix} U_{1,1} & \dots & U_{1,I} \\ \vdots & U_{k,i} & \vdots \\ U_{K,1} & \dots & U_{K,I} \end{bmatrix} \qquad V = \begin{bmatrix} V_{1,1} & \dots & V_{1,J} \\ \vdots & V_{k,j} & \vdots \\ V_{K,1} & \dots & V_{K,J} \end{bmatrix}$$

Now we can respectively represent the latent projections of user and item as $(U\mathbf{x})_{1\dots K}$ and $(V\mathbf{y})_{1\dots K}$ and hence use $\langle U\mathbf{x}, V\mathbf{y}\rangle = \mathbf{x}^T U^T V \mathbf{y}$ as a latent bilinear regressor.

2.2.3 Social Collaborative Filtering

There are essentially two general classes of MF methods applied to SCF that we discuss below. The first class can be termed as *social regularization* approaches in that they somehow constrain the latent projection represented by U.

There are two social regularization methods that directly constrain U for user i and k based on evidence $S_{i,k}$ of interaction between i and k. We call these methods:

- **Social regularization [Yang et al. 2011; Cui et al. 2011]**:

$$\sum_i \sum_{k \in friends(i)} \frac{1}{2}(S_{i,k} - \langle U_i, U_k \rangle)^2$$

- **Social spectral regularization [Ma et al. 2011; Li and Yeung 2009]**:

$$\sum_i \sum_{k \in friends(i)} \frac{1}{2} S^+_{i,k} \|U_i - U_k\|_2^2$$

- **SoRec regularization [Ma et al. 2008]**:

  The *SoRec* system [Ma et al. 2008] proposes a slight twist on social spectral regularization in that it learns a third $(N \times N)$ *interactions matrix* Z, and uses $U_i^T Z_k$ to predict user-user interaction preferences in the same way that standard CF uses V in $U_i^T V_j$ to predict user-item ratings. *SoRec* also uses a sigmoidal transform on the predictions:

$$\sum_i \sum_{k \in friends(i)} \frac{1}{2}(S_{i,k} - \sigma(\langle U_i, Z_k \rangle))^2$$

- **Social Trust Ensemble [Ma et al. 2009] (Non-spectral)**:

  The second class of SCF MF approaches represented by the single examplar of the *Social Trust Ensemble* can be termed as a *weighted average* approach since this approach simply composes a prediction for item j from a weighted average of a user i's predictio ns *as well as* their friends (k) predictions (as evidenced by the additional $\sum_k$ in the objective below).

$$\sum_{(i,j) \in D} \frac{1}{2}(R_{i,j} - \sigma(U_i^T V_j + \sum_k U_i^T V_k))^2$$

2.2.4 Tensor Factorization Methods

Tensor factorization (TF) methods can be used to learn latent models of interaction of 2 dimensions and higher. A dimension 2 TF method is simply standard MF. An example of a dimension 3 TF method is given by [Rendle et al. 2009] where recommendation of user-specific tags for an item are modeled with tags, user, and items each in one dimension. To date, TF methods have not been used for social recommendation, however, we draw on the idea of addition dimensions of latent learning in our co-preference regularization method.

In this chapter we have seen some of the different existing methods for CF and SCF. Each of these methods has its own weakness which we detailed in Chapter 1. Next, we discuss how we evaluate existing these CF and SCF algorithms, as well as the new SCF algorithms designed for this paper.

Chapter 3

Evaluation of Social Recommendation Systems

3.1 Facebook

Facebook is a social networking service that is currently the largest in the world. As of July 2011 it had more that 750 million active users. Users in Facebook create a profile and establish "friend" connections between users to establish their social network. Each user has a "wall" where they and their friends can make posts to. These posts can be links, photos, status updates, etc. Items that have been posted by a user can be "liked", shared, or commented upon by other users.

Figure 3.1: A link posted by the author that has been liked by three other users.

This paper seeks to find out how best to recommend links to individual users such that there is a high likelihood of them "liking" it. We do this by creating a Facebook application that recommends links to users everyday and the users could give their feedback on the links, whether they liked it or disliked it.

3.1.1 LinkR

Facebook allows applications to be developed that can be installed by their users. As part of this project, the LinkR Facebook application was developed. The functionalities of the LinkR application are:

1. Collect data that have been shared by users and their friends on Facebook.

2. Recommend links to the users daily.

3. Collect feedback from the users on whether they liked or disliked the recommendations.

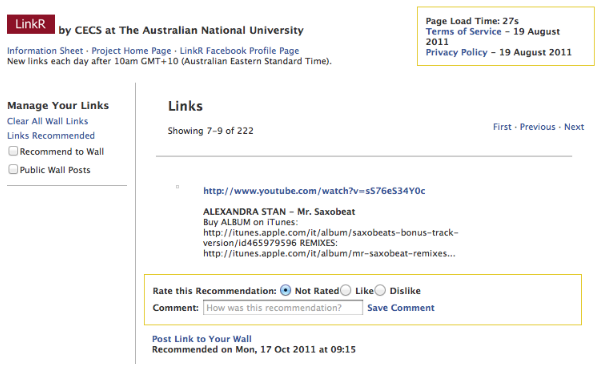


Figure 3.2: The LinkR application showing one of the recommendations.

The main developer of LinkR is Khoi-Nguyen Tran, a PhD student at the Australian National University. The algorithms it uses at the backend for recommendation were developed as part of this paper.

3.2 Dataset

Using the LinkR Facebook application developed for this project, we were able to gather data on 34,245 users and 407,887 links.[1]

3.2.1 User Data

Date that are used for the user features

- Gender
- Birthday
- *location_id*
- *hometown_id*
- Mapping of whether users $\mathbf{x}$ and $\mathbf{z}$ are friends.
- Interactions on Facebook between users $\mathbf{x}$ and $\mathbf{z}$. These interactions are used to build $Int_{\mathbf{x},\mathbf{z}}$.

[1] As of October 18, 2011, 12:15am

3.2.2 Link Data

Data that are used for the link features are:

- *id* of the user who posted the link.
- *id* of the user on whose wall the link was posted.
- Description of the link from the user who posted it.
- Link summary from the webpage.
- Number of times the link has been liked.
- Number of times the link has been shared.
- Number of comments posted on the link.
- List of users that have "liked" the link.

Additionally, links that have been recommended by the LinkR application have the following extra features:

- *id*'s of users who have clicked on the url.
- Optional "Like" or "Dislike" rating of the LinkR user on the link.

3.2.3 Implicit Dislikes

Outside of the "Dislike" ratings that we are able to get from the LinkR data, there is no other functionality within Facebook itself that allows users to explicitly define which link they do not like. Therefore, we need some way to infer disliked links during training. During training we consider links that were posted by the user's friends and which they have not likes as an evidence that they dislike a link. This is actually a big assumption as in a lot of cases given the nature of the Facebook news feed they may simply have not seen the link yet, and may actually like the link if they see it. Nevertheless, we find in our passive experiment and in live trial that this assumption is still useful.

3.3 Evaluation Metrics

We define True Positives (TP) to be the count of relevant items that were returned by the algorithm, False Positives (FP) to be the count of non-relevant items that were returned by the algorithm, True Negatives (TN) to be the count of non-relevant items that weren't returned by the algorithm, and False Negatives (FN) to be the non-relevant items that were returned by the algorithm.

Precision is a measure of what fraction of items returned by the algorithm were actually relevant.

$$Precision = \frac{TP}{TP + FP}$$

For some problems, results are returned as a ranked list. The position of an item in the list must also be evaluated, not just whether the item is in the returned list or not. A metric that does this is Average Precision, which computes the precision at every position in a ranked sequence of documents. k is the rank in a sequence of retrieved documents, n is the number of retrieved documents, and $P(k)$ is the precision at cut-off k in the list. $rel(k)$ is an indicator function equalling 1 if the item at position k is a relevant document, and 0 otherwise. The average precision is calculated as

$$AveP = \frac{\sum_{k=1}^{n}(P(k) \times rel(k))}{\text{number of relevant problems}}$$

The main metric we use in this paper is the mean average precision (MAP). Since we make a recommendation for each user, these recommendations can be viewed as a separate problem per user, and evaluate the average precision for each one. Getting the mean of all the average precisions gives us an effective metric for the entire recommendation system.

$$MAP = \frac{\sum_{u=1}^{U} AveP(u)}{\text{number of users}}$$

3.4 Training and Testing Issues

3.4.1 Training Data

Because of the sheer size of the Facebook data, it was impractical to run training and recommendations over the entire dataset. To keep the runtime of our experiments within reason, we used only the most recent four weeks of data for training the recommenders. This also helps alleviate some temporal aspects of the user's changing preferences, i.e., what the user liked last year may not be the same as what he or she likes this year. We also distinguish between the three types of link like/dislike data we can get from the dataset:

- ACTIVE: The explicit "Like" and "Dislike" rating that a LinkR user gives on links recommended by the LinkR application. In addition to this, a click by a user on a recommended link also counts as a like by that user on that particular link. Only LinkR users have this data.

- PASSIVE: The list of likes by users on links in the Facebook data and the inferred dislikes detailed above.

- UNION: Combination of the ACTIVE and PASSIVE data.

Figure 3.3: Screenshot of a recommendation made by the LinkR application with the rating and feedback options.

3.4.2 Live Online Recommendations

For the recommendations made to the LinkR application users, we select only links posted in the most recent two weeks that the user has not liked. We use only the links from the last two weeks because we consider recency to be a big issue. Older links have a greater chance of being about things that are outdated already, or worse, it may already be a broken link and not working anymore. We have settled on recommending three links per day to the LinkR users and according to the survey done at the end of the first trial, three links per day seems to be just the right number.

For the live trials, Facebook users who installed the LinkR application were randomly assigned one of four algorithms in each of the two trials. Users were not informed which algorithm was assigned to them to remove any bias. We distinguish our recommended links into two major classes, links that were posted by the LinkR user's friends and links that were posted by users other than the LinkR user's friends. The LinkR users were encouraged to rate the links that were recommended to them, and even provide feedback comments on the specific links. In turn these ratings became part of the training data for the recommendation algorithms, and thus was used to improve the performance of the algorithms over time. Based on the user feedback, we filtered out non-English links and links without any descriptions from the recommendations to prevent user annoyance.

At the end of the first trial, we conducted a user survey with the LinkR users to find out how satisfied they were with the recommendations they were getting.

3.4.3 Test Data

Similar to our selection for training data, the test data used for our passive experiment also uses only the most recent 4 weeks of data. We distinguish the test data into the following classes:

- FB-USER-PASSIVE: The PASSIVE like/dislike data for all Facebook users in the dataset.

- APP-USER-PASSIVE: The PASSIVE like/dislike data for only the LinkR application users.
- APP-USER-ACTIVE-FRIENDS: The ACTIVE like/dislike data for the LinkR users, but only for friend recommended links.
- APP-USER-ACTIVE-NON-FRIENDS: The ACTIVE like/dislike data for the LinkR users, but only for non-friend recommended links.
- APP-USER-ACTIVE-ALL: The entire active like/dislike data for the LinkR users.

During passive experiments, we simply select which combination of training data and testing data to use. This helped us see which training-test data combination best reflected the results of the live trials. In cases where training and testing data overlap, i.e., training on PASSIVE and testing on APP-USER-PASSIVE, we get a random 20% subset of the training data per user for testing. These links are then removed from the training data to ensure that there are no common links between the training data and the test data.

Chapter 4

Comparison of Existing Recommender Systems

4.1 Objective components

We take a composable approach to collaborative filtering (CF) systems where a (social) CF minimization objective Obj is composed of sums of one or more objective components:

$$Obj = \sum_i \lambda_i Obj_i \tag{4.1}$$

Because each objective may be weighted differently, a weighting term $\lambda_i \in \mathbb{R}$ for each component that should be optimized via cross-validation.

Most target predictions are binary classification-based ($\{0, 1\}$), therefore in the objectives a sigmoidal transform

$$\sigma(o) = \frac{1}{1 + e^{-o}} \tag{4.2}$$

of regressor outputs $o \in \mathbb{R}$ is used to squash it to the range $[0, 1]$. In places where the σ transform may be optionally included, this is written as $[\sigma]$.

4.1.1 Matchbox Matrix Factorization

The basic objective function we use for our MF models is the Matchbox [Stern et al. 2009] model:

$$\sum_{(\mathbf{x},\mathbf{y}) \in D} \frac{1}{2}(R_{\mathbf{x},\mathbf{y}} - [\sigma]\mathbf{x}^T U^T V y)^2 \tag{4.3}$$

4.1.2 L2 Regularization

To help in generalization, it is important to regularize the free parameters U and V to prevent overfitting in the presence of sparse data. This can be done with the L_2

regularizer that models a prior of 0 on the parameters. The objective components for the L2 regularizers are

$$\frac{1}{2}\|U\|_{\text{Fro}}^2 = \frac{1}{2}\,\text{tr}(U^T U) \tag{4.4}$$

$$\frac{1}{2}\|V\|_{\text{Fro}}^2 = \frac{1}{2}\,\text{tr}(V^T V) \tag{4.5}$$

4.1.3 Social Regularization

The social aspect of social recommendation is implemented as a regularizer on the user matrix. What this objective component does is constrain users with a high similarity rating to have the same values in the latent feature space. This models the assumption that users who are similar socially should have the same preferences for items.

This method is an extension of existing SCF techniques [Yang et al. 2011; Cui et al. 2011] that constrain the latent space to enforce users to have similar preferences latent representations when they interact heavily. Like Matchbox which extends regular matrix factorization methods by making use of user and link features, our Social Regularization method incorporates user features to learn user similarities.

$$\begin{aligned}
&\sum_{\mathbf{x}} \sum_{\mathbf{z}\in friends(\mathbf{x})} \frac{1}{2}(S_{\mathbf{x},\mathbf{z}} - \langle U\mathbf{x}, U\mathbf{z}\rangle)^2 \\
&\quad = \sum_{\mathbf{x}} \sum_{\mathbf{z}\in friends(\mathbf{x})} \frac{1}{2}(S_{\mathbf{x},\mathbf{z}} - \mathbf{x}^T U^T U\mathbf{z})^2
\end{aligned} \tag{4.6}$$

4.1.4 Derivatives

We seek to optimize sums of the above objectives and will use gradient descent for this purpose.

For the overall objective, the partial derivative w.r.t. parameters $\mathbf{a}$ are as follows:

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{a}} Obj &= \frac{\partial}{\partial \mathbf{a}} \sum_i \lambda_i Obj_i \\
&= \sum_i \lambda_i \frac{\partial}{\partial \mathbf{a}} Obj_i
\end{aligned}$$

Previously we noted that that we may want to transform some of the regressor outputs $o[\cdot]$ using $\sigma(o[\cdot])$. This is convenient for our partial derivatives as

$$\frac{\partial}{\partial \mathbf{a}}\sigma(o[\cdot]) = \sigma(o[\cdot])(1 - \sigma(o[\cdot]))\frac{\partial}{\partial \mathbf{a}} o[\cdot]. \tag{4.7}$$

Hence anytime a $[\sigma(o[\cdot])]$ is optionally introduced in place of $o[\cdot]$, we simply insert $[\sigma(o[\cdot])(1-\sigma(o[\cdot]))]$ in the corresponding derivatives below.[1]

Before we proceed to our objective gradients, we define abbreviations for two useful vectors:

$$\begin{aligned} \mathbf{s} &= U\mathbf{x} \qquad \mathbf{s}_k = (U\mathbf{x})_k;\ k = 1\ldots K \\ \mathbf{t} &= V\mathbf{y} \qquad \mathbf{t}_k = (V\mathbf{y})_k;\ k = 1\ldots K \end{aligned}$$

Now we proceed to derivatives for the previously defined primary objective components:

- **Matchbox Matrix Factorization**: Here we define alternating partial derivatives between U and V, holding one constant and taking the derivative w.r.t. the other:[2]

$$\begin{aligned} \frac{\partial}{\partial U} Obj_{pmcf} &= \frac{\partial}{\partial U} \sum_{(\mathbf{x},\mathbf{y})\in D} \frac{1}{2} \left(\underbrace{(R_{\mathbf{x},\mathbf{y}} - [\sigma]\, \overbrace{x^T U^T V \mathbf{y}}^{o_{\mathbf{x},\mathbf{y}}})}_{\delta_{\mathbf{x},\mathbf{y}}} \right)^2 \\ &= \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} \frac{\partial}{\partial U} - [\sigma]\mathbf{x}^T U^T \mathbf{t} \\ &= - \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} [\sigma(o_{\mathbf{x},\mathbf{y}})(1-\sigma(o_{\mathbf{x},\mathbf{y}}))]\mathbf{t}\mathbf{x}^T \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial V} Obj_{pmcf} &= \frac{\partial}{\partial V} \sum_{(\mathbf{x},\mathbf{y})\in D} \frac{1}{2} \left(\underbrace{(R_{\mathbf{x},\mathbf{y}} - [\sigma]\, \overbrace{x^T U^T V \mathbf{y}}^{o_{\mathbf{x},\mathbf{y}}})}_{\delta_{\mathbf{x},\mathbf{y}}} \right)^2 \\ &= \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} \frac{\partial}{\partial V} - [\sigma]\mathbf{s}^T V \mathbf{y} \\ &= - \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} [\sigma(o_{\mathbf{x},\mathbf{y}})(1-\sigma(o_{\mathbf{x},\mathbf{y}}))]\mathbf{s}\mathbf{y}^T \end{aligned}$$

For the regularization objective components, the derivatives are:

- L_2 U **regularization**:

$$\begin{aligned} \frac{\partial}{\partial U} Obj_{ru} &= \frac{\partial}{\partial U} \frac{1}{2} \operatorname{tr}(U^T U) \\ &= U \end{aligned}$$

[1] We note that our experiments using the sigmoidal transform in objectives with $[0,1]$ predictions do not generally demonstrate a clear advantage vs. the omission of this transform as originally written (although they do not demonstrate a clear disadvantage either).

[2] We will use this method of alternation for all objective components that involve bilinear terms.

- L_2 V **regularization**:

$$\begin{aligned}\frac{\partial}{\partial V} Obj_{rv} &= \frac{\partial}{\partial V} \frac{1}{2} \operatorname{tr}(V^T V) \\ &= V\end{aligned}$$

- **Social regularization**:

$$\begin{aligned}\frac{\partial}{\partial U} Obj_{rs} &= \frac{\partial}{\partial U} \sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} \frac{1}{2} \left(\underbrace{S_{\mathbf{x},\mathbf{z}} - \mathbf{x}^T U^T U \mathbf{z}}_{\delta_{\mathbf{x},\mathbf{y}}} \right)^2 \\ &= \sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} \delta_{\mathbf{x},\mathbf{y}} \frac{\partial}{\partial U} - \mathbf{x}^T U^T U \mathbf{z} \\ &= -\sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} \delta_{\mathbf{x},\mathbf{y}} U (\mathbf{x}\mathbf{z}^T + \mathbf{z}\mathbf{x}^T)\end{aligned}$$

Hence, for any choice of primary objective and one or more regularizers, we simply add the derivatives for U and/or V according to (4.7).

4.2 Algorithms

The recommendations used for the first trial were

1. ***k*-Nearest Neighbor**
2. **Support Vector Machines**
3. **Matchbox**: Matchbox MF + L2 Regularization
4. **Social Matchbox**: Matchbox MF + Social Regularization + L2 Regularization

Social Matchbox uses the Social Regularizion method to incorporate the social aspect of the data. SVM, Matchbox and Nearest Neighbors do not make use of any social information and are collaborative filtering recommenders.

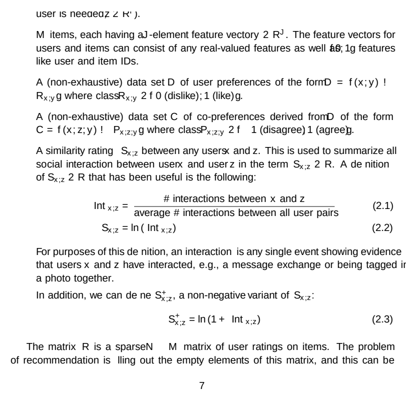
4.3 Online Results

The first live user trial was run from August 1 to October 13, though we include only data starting from August 25 in our results because that is when the parameters of the algorithms were stabilized.. The algorithms were randomly distributed among the 106 users who installed the LinkR application. Each user was recommended 3 links everyday and they were able to rate the links on whether they 'Liked' or 'Disliked' it.

Results shown are the number like ratings and the number of dislike ratings normalized by the total number of ratings (likes + dislikes) per algorithm. Social Matchbox

Algorithm	Users
Social Matchbox	26
Matchbox	26
SVM	28
Nearest Neighbor	28

Table 4.1: Number of Users Assigned per Algorithm.


user is needed $z \in R^{J}$).

M items, each having a J-element feature vector $y \in R^J$. The feature vectors for users and items can consist of any real-valued features as well as $\{0, 1\}$ features like user and item IDs.

A (non-exhaustive) data set D of user preferences of the form $D = \{(x; y) \rightarrow R_{x;y}\}$ where class $R_{x;y} \in \{0 \text{ (dislike)}; 1 \text{ (like)}\}$.

A (non-exhaustive) data set C of co-preferences derived from D of the form $C = \{(x; z; y) \rightarrow P_{x;z;y}\}$ where class $P_{x;z;y} \in \{-1 \text{ (disagree)}; 1 \text{ (agree)}\}$.

A similarity rating $S_{x;z}$ between any users x and z. This is used to summarize all social interaction between user x and user z in the term $S_{x;z} \in R$. A definition of $S_{x;z} \in R$ that has been useful is the following:

$$Int_{x;z} = \frac{\text{\# interactions between } x \text{ and } z}{\text{average \# interactions between all user pairs}} \quad (2.1)$$

$$S_{x;z} = \ln(Int_{x;z}) \quad (2.2)$$

For purposes of this definition, an interaction is any single event showing evidence that users x and z have interacted, e.g., a message exchange or being tagged in a photo together.

In addition, we can define $S^+_{x;z}$, a non-negative variant of $S_{x;z}$:

$$S^+_{x;z} = \ln(1 + Int_{x;z}) \quad (2.3)$$

The matrix R is a sparse $N \times M$ matrix of user ratings on items. The problem of recommendation is filling out the empty elements of this matrix, and this can be

7

Figure 4.1: Results of the online live trials. Social Matchbox was found to be the best performing algorithm.

was the best performing algorithm in the first trial and in fact was the only algorithm to get receive more like ratings than dislike ratings.

We also look at the algorithms with the results split between friend links and non-friend links recommendations. Again, the results shown are the number of likes or dislikes from friends or non-friends normalized by the total number of ratings on those type of recommendations. All four algorithms experiences a significant performance drop in the number of likes when it came to recommending non-friend links. Aside from liking and disliking a link just from the quality of the links being recommended, it seems that users are also more likely to like a link simply because a friend had posted it and more likely to dislike it just because it came from a stranger.

4.4 Offline Results

The goal of the offline experiments was to see how best to reproduce the results of the live experiments offline. The algorithms are being evaluated using the Mean Average Precision metric. he offline experiments were also used to tune the parameters of the different algorithms. When training on the UNION dataset, we can find the same general worsening of performance between the results of testing on APP-USER-ACTIVE-FRIENDS and APP-USER-ACTIVE-NON-FRIENDS. Additionally, when testing on the APP-USER-ACTIVE-ALL, the training/test data combination most similar to the online setup, we find that Social Matchbox slightly beat the other algorithms.

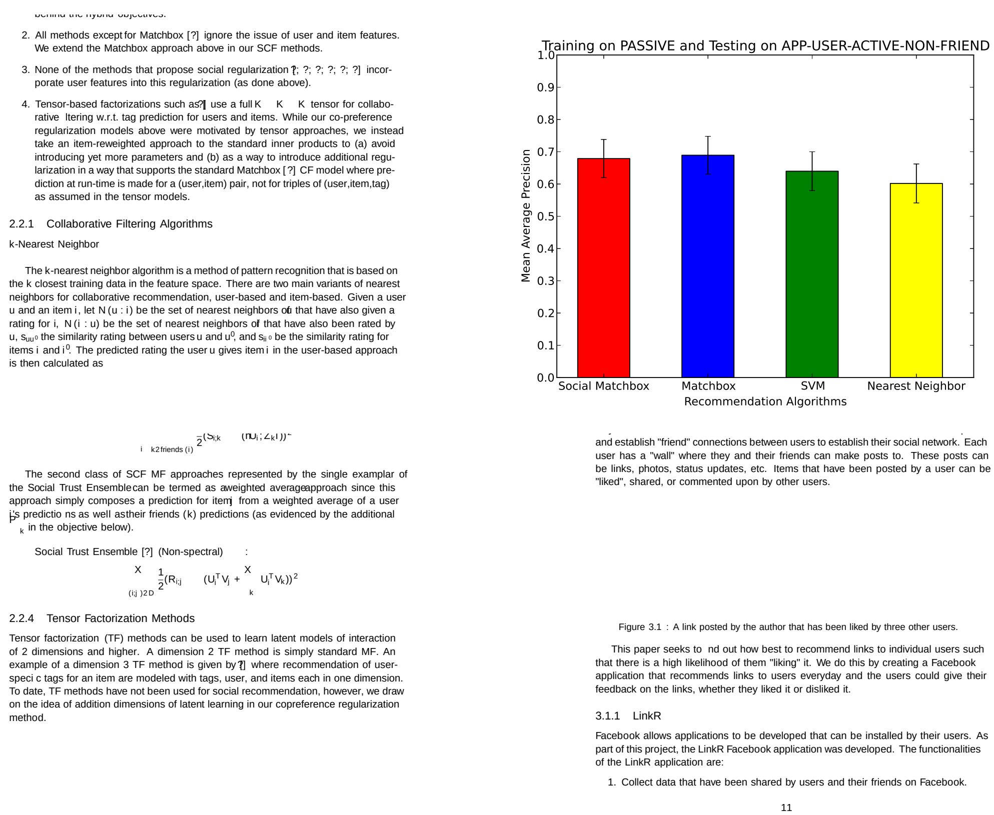


Figure 4.2: Results of the online live trials, split between friends and non-friends. There is a significant drop in performance between recommending friend links and recommending non-friend links.

4.5 Survey Results

Near the end of the first trial, the LinkR users were invited to answer a survey regarding their experiences with the recommendations they were getting. They were asked a number of questions, with the following pertaining to the quality of the recommendations:

- Do you find that ANU LinkR recommends interesting links that you may not have otherwise seen?
- Do you feel that ANU LinkR has adapted to your preferences since you first started using it?
- How relevant are the daily recommended links?
- Overall, how satisfied are you with LinkR?

They gave their answers to each question as an integer rating with range $[1-5]$, with a higher value being better. Their answers were grouped together according to the recommendation algorithm that was assigned to them, and the averages per algorithm are below.

One more, we see that Social Matchbox achieved higher scores than the other recommendation algorithms, in all four questions. The results of the survey reflected the results in the online live trial and confirms that Social Matchbox was the best recommendation algorithm in the first trial.

At the end of the first trial, we have seen that the Social Matchbox its Social Regularization technique does indeed outperform existing collaborative filtering recommenders. In the next chapter, we discuss new techniques for incorporating social information and show how they improve on Social Matchbox.

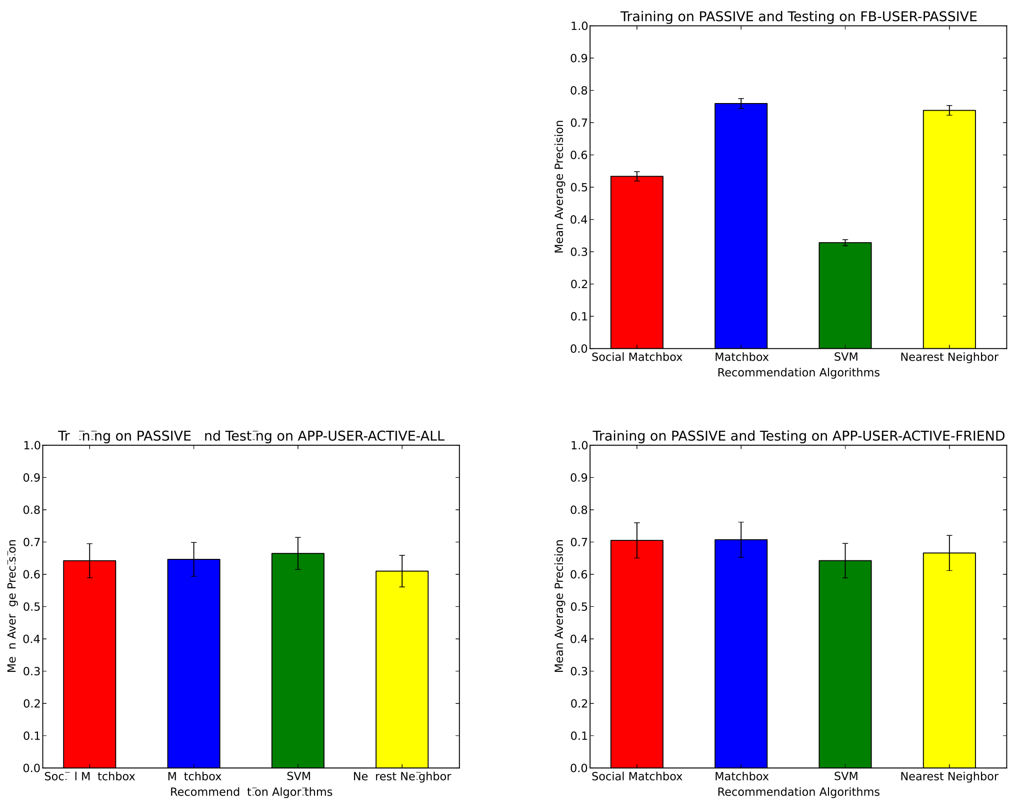
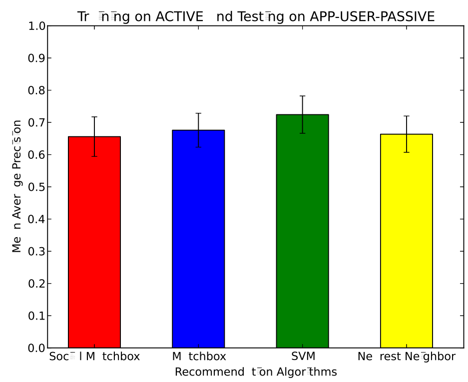


Figure 4.3: Results of training on PASSIVE data



recommendation system.

$$MAP = \frac{\sum_{u=1}^{U} AveP(u)}{\text{number of users}}$$

3.4 Training and Testing Issues

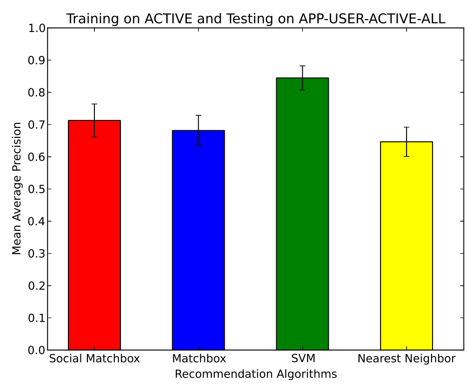
3.4.1 Training Data

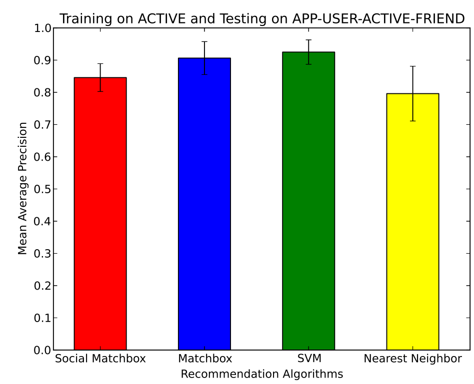
Because of the sheer size of the Facebook data, it was impractical to run training and recommendations over the entire dataset. To keep the runtime of our experiments within reason, we used only the most recent four weeks of data for training the recommenders. This also helps alleviate some temporal aspects of the user's changing preferences, i.e., what the user liked last year may not be the same as what he or she likes this year. We also distinguish between the three types of link like/dislike data we can get from the dataset:

ACTIVE: The explicit "Like" and "Dislike" rating that a LinkR user gives on links recommended by the LinkR application. In addition to this, a click by a user on a recommended link also counts as a like by that user on that particular link. Only LinkR users have this data.

PASSIVE: The list of likes by users on links in the Facebook data and the inferred dislikes detailed above.

UNION: Combination of the ACTIVE and PASSIVE data.





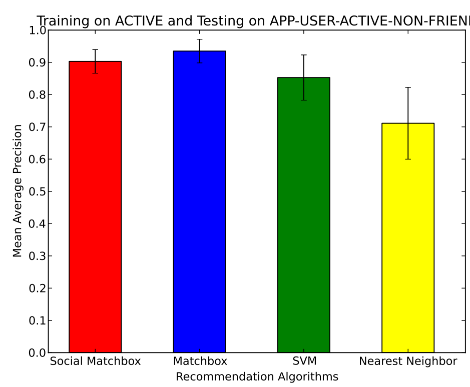


Figure 4.4: Results of training on ACTIVE data

from the last two weeks because we consider recency to be a big issue. Older links have a greater chance of being about things that are outdated already, or worse, it may already be a broken link and not working anymore. We have settled on recommending three links per day to the LinkR users and according to the survey done at the end of the rst trial, three links per day seems to be just the right number.

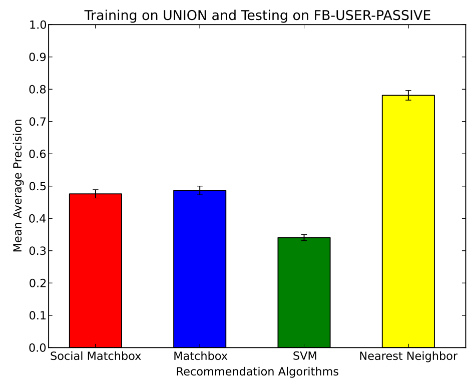
For the live trials, Facebook users who installed the LinkR application were randomly assigned one of four algorithms in each of the two trials. Users were not informed which algorithm was assigned to them to remove any bias. We distinguish our recommended links into two major classes, links that were posted by the LinkR user's friends and links that were posted by users other than the LinkR user's friends. The LinkR users were encouraged to rate the links that were recommended to them, and even provide feedback comments on the speci c links. In turn these ratings became part of the training data for the recommendation algorithms, and thus was used to improve the performance of the algorithms over time. Based on the user feedback, we ltered out non-English links and links without any descriptions from the recommendations to prevent user annoyance.

At the end of the rst trial, we conducted a user survey with the LinkR users to nd out how satis ed they were with the recommendations they were getting.

3.4.3 Test Data

Similar to our selection for training data, the test data used for our passive experiment also uses only the most recent 4 weeks of data. We distinguish the test data into the following classes:

FB-USER-PASSIVE: The PASSIVE like/dislike data for all Facebook users in the dataset.



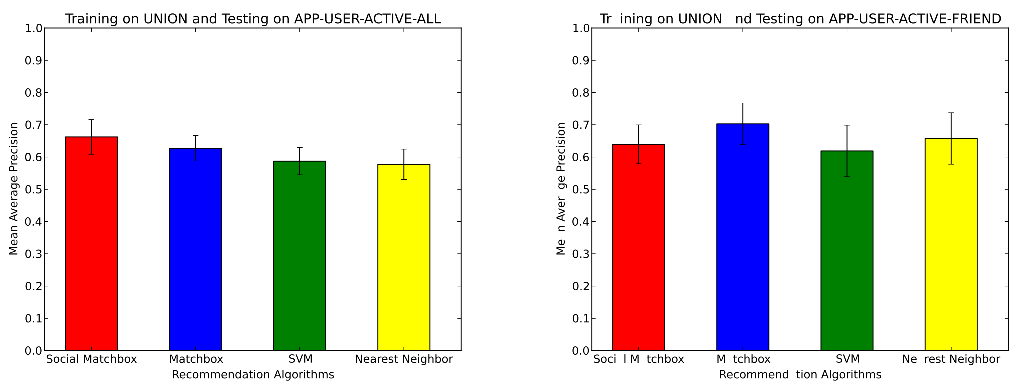


Figure 4.5: Results of training on UNION data. When testing on APP-USER-ACTIVE-ALL, we find that Social Matchbox was again the best recommendation algorithm. Training on UNION and testing on APP-USER-ACTIVE-ALL is the training/test data combination that is most similar to the online setup.

components:

$$\mathrm{Obj} = \sum_i \lambda_i \mathrm{Obj}_i \quad (4.1)$$

Because each objective may be weighted di erently, a weighting term $\lambda_i \in \mathbb{R}$ for each component that should be optimized via cross-validation.

Most target predictions are binary classi cation-based ($\in \{0;1\}$), therefore in the objectives a sigmoidal transform

$$\sigma(o) = \frac{1}{1+e^{-o}} \quad (4.2)$$

of regressor outputs $o \in \mathbb{R}$ is used to squash it to the range $[0;1]$. In places where the σ transform may be optionally included, this is written as $[\sigma]$.

4.1.1 Matchbox Matrix Factorization

The basic objective function we use for our MF models is the Matchbox [?] model:

$$\sum_{(x;y)\in D} \frac{1}{2}(R_{x;y} - [\sigma]x^T U^T V y)^2 \quad (4.3)$$

4.1.2 L2 Regularization

To help in generalization, it is important to regularize the free parameters U and V to prevent over tting in the presence of sparse data. This can be done with the L_2

17

the latent space to enforce users to have similar preferences latent representations when they interact heavily. Like Matchbox which extends regular matrix factorization methods by making use of user and link features, our Social Regularization method incorporates user features to learn user similarities.

$$\sum_x \sum_{z\in \mathrm{friends}(x)} \frac{1}{2}(S_{x;z} - \langle Ux; Uz\rangle)^2 = \sum_x \sum_{z\in \mathrm{friends}(x)} \frac{1}{2}(S_{x;z} - x^T U^T U z)^2 \quad (4.6)$$

4.1.4 Derivatives

We seek to optimize sums of the above objectives and will use gradient descent for this purpose.

For the overall objective, the partial derivative w.r.t. parameters a are as follows:

$$\frac{\partial}{\partial a}\mathrm{Obj} = \frac{\partial}{\partial a}\sum_i \lambda_i \mathrm{Obj}_i = \sum_i \lambda_i \frac{\partial}{\partial a}\mathrm{Obj}_i$$

Previously we noted that that we may want to transform some of the regressor outputs $o[\cdot]$ using $\sigma(o[\cdot])$. This is convenient for our partial derivatives as

$$\frac{\partial}{\partial a}\sigma(o[\cdot]) = \sigma(o[\cdot])(1 - \sigma(o[\cdot]))\frac{\partial}{\partial a}o[\cdot]: \quad (4.7)$$

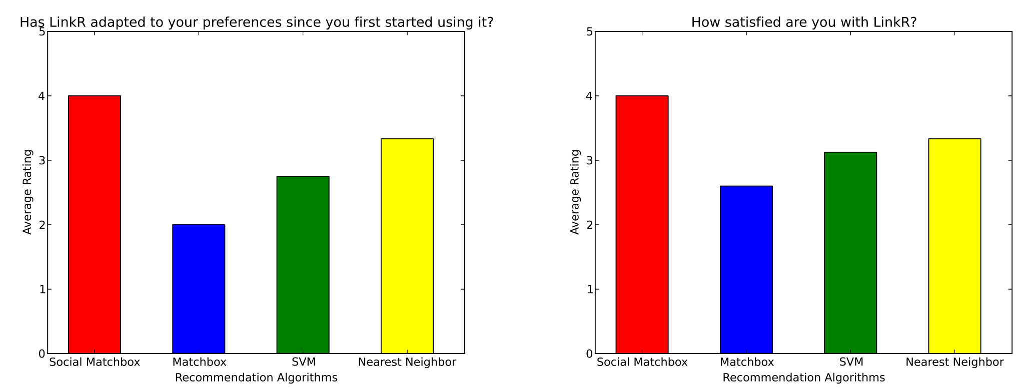


Figure 4.6: Results of the user survey after the first trial. The survey answers from the users reflect the online results that Social Matchbox was the best recommendation algorithm in this trial.

Chapter 5

New Algorithms for Social Recommendation

After studying the results of the first trial, we made use of what we learned to come up with new algorithms for social collaborative filtering. Our goal in designing these algorithms was to address deficiencies in current SCF methods that we pointed out in the Introduction chapter:

- Non-feature-based user similarity
- Modeling direct user-user information diffusion
- Restricted common interests

5.1 New Objective Components

Again, the new algorithms each form a component of a minimization objective Obj which is composed of sums of one or more objective components:

$$Obj = \sum_i \lambda_i Obj_i \tag{5.1}$$

A sigmoidal transform

$$\sigma(o) = \frac{1}{1 + e^{-o}} \tag{5.2}$$

of regressor outputs $o \in \mathbb{R}$ is used to squash the outputs to the range $[0, 1]$. In places where the σ transform may be optionally included, this is written as $[\sigma]$.

5.1.1 Hybrid Objective

As specified in Chapter 1, one weakness of MF methods is that they cannot model joint features over user and items, and hence the cannot model direct user-user information diffusion. We fix this by introducing another objective component in addition to the standard MF objective, and this component serves as a simple linear regressor for such information diffusion observations. The resulting hybrid objective component becomes a combination of latent MF and linear regression objectives.

We make use of the $\mathbf{f}_{\mathbf{x},\mathbf{y}}$ features to make the linear regressor. Using $\langle \cdot, \cdot \rangle$ to denote an inner product, we define a weight vector $\mathbf{w} \in \mathbb{R}^F$ such that $\langle \mathbf{w}, \mathbf{f}_{\mathbf{x},\mathbf{y}} \rangle = \mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}}$ is the prediction of the system. The objective of the linear regression component is therefore

$$\sum_{(\mathbf{x},\mathbf{y}) \in D} \frac{1}{2}(R_{\mathbf{x},\mathbf{y}} - [\sigma]\mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}})^2$$

We combine the output of the linear regression objective with the Matchbox output $[\sigma]\mathbf{x}^T U^T V y$, to get a hybrid objective component. The full objective function for this hybrid model is

$$\sum_{(\mathbf{x},\mathbf{y}) \in D} \frac{1}{2}(R_{\mathbf{x},\mathbf{y}} - [\sigma]\mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}} - [\sigma]\mathbf{x}^T U^T V y)^2 \tag{5.3}$$

5.1.2 Social Spectral Regularization

As we did with the Social Regularization method in Section 4.1.3, we build on ideas used in Matchbox [Stern et al. 2009] to extend social spectral regularization [Ma et al. 2011; Li and Yeung 2009] by incorporating user features into the objective.

$$\begin{aligned}
&\sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} \frac{1}{2} S^+_{\mathbf{x},\mathbf{z}} \|U\mathbf{x} - U\mathbf{z}\|_2^2 \\
&= \sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} \frac{1}{2} S^+_{\mathbf{x},\mathbf{z}} \|U(\mathbf{x} - \mathbf{z})\|_2^2 \\
&= \sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} \frac{1}{2} S^+_{\mathbf{x},\mathbf{z}} (\mathbf{x} - \mathbf{z})^T U^T U (\mathbf{x} - \mathbf{z})
\end{aligned} \tag{5.4}$$

Note: standard spectral regularization assumes $S^+_{\mathbf{x},\mathbf{z}} \in [0, 1]$; however we may also want to try $S_{\mathbf{x},\mathbf{z}}$ since a negative value actively encourages the latent spaces to oppose each other, which may be desired.

5.1.3 Social Co-preference Regularization

A crucial aspect missing from other SCF methods is that while two users may not be globally similar or opposite in their preferences, there may be sub-areas of their interests which can be correlated to each other. For example, two friends may have similar interests concerning music, but different interests concerning politics. The social co-preference regularizers aim to learn such selective co-preferences. The motivation is to constrain users $\mathbf{x}$ and $\mathbf{z}$ who have similar or opposing preferences to be similar or opposite in the same latent latent space relevant to item $\mathbf{y}$.

We use $\langle \cdot, \cdot \rangle_{\bullet}$ to denoe a reweighted inner product. The objective component for social co-preference regularization along with its expanded form is

$$\begin{aligned}
&\sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \frac{1}{2} (P_{\mathbf{x},\mathbf{z},\mathbf{y}} - \langle U\mathbf{x}, U\mathbf{z} \rangle_{V\mathbf{y}})^2 \\
&\qquad = \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \frac{1}{2} (P_{\mathbf{x},\mathbf{z},\mathbf{y}} - \mathbf{x}^T U^T \operatorname{diag}(V\mathbf{y}) U\mathbf{z})^2
\end{aligned} \tag{5.5}$$

5.1.4 Social Co-preference Spectral Regularization

This is the same as the social co-preference regularization above, except that it uses the spectral regularizer format for learning the co-preferences.

We use $\| \cdot \|_{2,\bullet}$ to denote a re-weighted L_2 norm. The objective component for social co-preference spectral regularization along with its expanded form is

$$\begin{aligned}
&\sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \frac{1}{2} P_{\mathbf{x},\mathbf{z},\mathbf{y}} \| U\mathbf{x} - U\mathbf{z} \|_{2,V\mathbf{y}}^2 \\
&\qquad = \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \frac{1}{2} P_{\mathbf{x},\mathbf{z},\mathbf{y}} \| U(\mathbf{x} - \mathbf{z}) \|_{2,V\mathbf{y}}^2 \\
&\qquad = \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \frac{1}{2} P_{\mathbf{x},\mathbf{z},\mathbf{y}} (\mathbf{x} - \mathbf{z})^T U^T \operatorname{diag}(V\mathbf{y}) U(\mathbf{x} - \mathbf{z})
\end{aligned} \tag{5.6}$$

5.1.5 Derivatives

As before, we seek to optimize sums of the above objectives and will use gradient descent for this purpose. We again use the following useful abbreviations:

$$\begin{aligned}
\mathbf{s} &= U\mathbf{x} \qquad & \mathbf{s}_k &= (U\mathbf{x})_k;\ k = 1 \ldots K \\
\mathbf{t} &= V\mathbf{y} \qquad & \mathbf{t}_k &= (V\mathbf{y})_k;\ k = 1 \ldots K
\end{aligned}$$

The derivatives for the linear CBF and hybrid objective functions, as well as the new social regularizers are

- **Explicit Linear CBF**:

$$
\begin{aligned}
\frac{\partial}{\partial \mathbf{w}} Obj_{pcbf} &= \frac{\partial}{\partial \mathbf{w}} \sum_{(\mathbf{x},\mathbf{y})\in D} \frac{1}{2} \left(\underbrace{(R_{\mathbf{x},\mathbf{y}} - [\sigma]\, \overbrace{\mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}}}^{o_{\mathbf{x},\mathbf{y}}})}_{\delta_{\mathbf{x},\mathbf{y}}} \right)^2 \\
&= \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} \frac{\partial}{\partial \mathbf{w}} - [\sigma] \mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}} \\
&= - \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} [\sigma(o_{\mathbf{x},\mathbf{y}})(1 - \sigma(o_{\mathbf{x},\mathbf{y}}))] \mathbf{f}_{\mathbf{x},\mathbf{y}}
\end{aligned}
$$

- **Hybrid**:

$$
\begin{aligned}
\frac{\partial}{\partial \mathbf{w}} Obj_{phy} &= \frac{\partial}{\partial \mathbf{w}} \sum_{(\mathbf{x},\mathbf{y})\in D} \frac{1}{2} \left(\underbrace{R_{\mathbf{x},\mathbf{y}} - [\sigma]\, \overbrace{\mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}}}^{o^1_{\mathbf{x},\mathbf{y}}} - [\sigma] \mathbf{x}^T U^T V \mathbf{y}}_{\delta_{\mathbf{x},\mathbf{y}}} \right)^2 \\
&= \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} \frac{\partial}{\partial \mathbf{w}} - [\sigma] \mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}} \\
&= - \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} [\sigma(o^1_{\mathbf{x},\mathbf{y}})(1 - \sigma(o^1_{\mathbf{x},\mathbf{y}}))] \mathbf{f}_{\mathbf{x},\mathbf{y}}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial}{\partial U} Obj_{phy} &= \frac{\partial}{\partial U} \sum_{(\mathbf{x},\mathbf{y})\in D} \frac{1}{2} \left(\underbrace{R_{\mathbf{x},\mathbf{y}} - [\sigma] \mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}} - [\sigma]\, \overbrace{\mathbf{x}^T U^T V \mathbf{y}}^{o^2_{\mathbf{x},\mathbf{y}}}}_{\delta_{\mathbf{x},\mathbf{y}}} \right)^2 \\
&= \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} \frac{\partial}{\partial U} - [\sigma] \mathbf{x}^T U^T V \mathbf{y} \\
&= - \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} [\sigma(o^2_{\mathbf{x},\mathbf{y}})(1 - \sigma(o^2_{\mathbf{x},\mathbf{y}}))] \mathbf{t} \mathbf{x}^T \\
\frac{\partial}{\partial V} Obj_{phy} &= \frac{\partial}{\partial V} \sum_{(\mathbf{x},\mathbf{y})\in D} \frac{1}{2} \left(\underbrace{R_{\mathbf{x},\mathbf{y}} - [\sigma] \mathbf{w}^T \mathbf{f}_{\mathbf{x},\mathbf{y}} - [\sigma]\, \overbrace{\mathbf{x}^T U^T V \mathbf{y}}^{o^2_{\mathbf{x},\mathbf{y}}}}_{\delta_{\mathbf{x},\mathbf{y}}} \right)^2 \\
&= \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} \frac{\partial}{\partial V} - [\sigma] \mathbf{x}^T U^T V \mathbf{y} \\
&= - \sum_{(\mathbf{x},\mathbf{y})\in D} \delta_{\mathbf{x},\mathbf{y}} [\sigma(o^2_{\mathbf{x},\mathbf{y}})(1 - \sigma(o^2_{\mathbf{x},\mathbf{y}}))] \mathbf{s} \mathbf{y}^T
\end{aligned}
$$

- **Social spectral regularization**:

$$
\begin{aligned}
\frac{\partial}{\partial U} Obj_{rss} &= \frac{\partial}{\partial U} \sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} \frac{1}{2} S^{+}_{\mathbf{x},\mathbf{z}} (\mathbf{x}-\mathbf{z})^T U^T U (\mathbf{x}-\mathbf{z}) \\
&= \sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} \frac{1}{2} S^{+}_{\mathbf{x},\mathbf{z}} U((\mathbf{x}-\mathbf{z})(\mathbf{x}-\mathbf{z})^T + (\mathbf{x}-\mathbf{z})(\mathbf{x}-\mathbf{z})^T) \\
&= \sum_{\mathbf{x}} \sum_{\mathbf{z} \in friends(\mathbf{x})} S^{+}_{\mathbf{x},\mathbf{z}} U (\mathbf{x}-\mathbf{z})(\mathbf{x}-\mathbf{z})^T
\end{aligned}
$$

Before we proceed to the final derivatives, we define one additional vector abbreviation:

$$
\mathbf{r} = U\mathbf{z} \qquad \mathbf{r}_k = (U\mathbf{z})_k;\ k = 1 \ldots K.
$$

- **Social co-preference regularization**:

$$
\begin{aligned}
\frac{\partial}{\partial U} Obj_{rsc} &= \frac{\partial}{\partial U} \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \frac{1}{2} \Bigg(\underbrace{P_{\mathbf{x},\mathbf{z},\mathbf{y}} - \mathbf{x}^T U^T \operatorname{diag}(V\mathbf{y}) U \mathbf{z}}_{\delta_{\mathbf{x},\mathbf{z},\mathbf{y}}} \Bigg)^2 \\
&= \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \delta_{\mathbf{x},\mathbf{z},\mathbf{y}} \frac{\partial}{\partial U} - \mathbf{x}^T U^T \operatorname{diag}(V\mathbf{y}) U \mathbf{z} \\
&= - \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \delta_{\mathbf{x},\mathbf{z},\mathbf{y}} (\operatorname{diag}(V\mathbf{y})^T U \mathbf{x}\mathbf{z}^T + \operatorname{diag}(V\mathbf{y}) U \mathbf{z}\mathbf{x}^T) \\
&= - \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \delta_{\mathbf{x},\mathbf{z},\mathbf{y}} \operatorname{diag}(V\mathbf{y}) U (\mathbf{x}\mathbf{z}^T + \mathbf{z}\mathbf{x}^T)
\end{aligned}
$$

In the following, $\circ$ is the Hadamard elementwise product:

$$
\begin{aligned}
\frac{\partial}{\partial V} Obj_{rsc} &= \frac{\partial}{\partial V} \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \frac{1}{2} (P_{\mathbf{x},\mathbf{z},\mathbf{y}} - \mathbf{x}^T U^T \operatorname{diag}(V\mathbf{y}) U \mathbf{z})^2 \\
&= \frac{\partial}{\partial V} \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \frac{1}{2} \Bigg(\underbrace{P_{\mathbf{x},\mathbf{z},\mathbf{y}} - (\overbrace{U\mathbf{x}}^{\mathbf{s}} \circ \overbrace{U\mathbf{z}}^{\mathbf{r}})^T V \mathbf{y}}_{\delta_{\mathbf{x},\mathbf{z},\mathbf{y}}} \Bigg)^2 \\
&= \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \delta_{\mathbf{x},\mathbf{z},\mathbf{y}} \frac{\partial}{\partial V} - (\mathbf{s} \circ \mathbf{r})^T V \mathbf{y} \\
&= - \sum_{(\mathbf{x},\mathbf{z},\mathbf{y}) \in C} \delta_{\mathbf{x},\mathbf{z},\mathbf{y}} (\mathbf{s} \circ \mathbf{r}) \mathbf{y}^T
\end{aligned}
$$

- **Social co-preference spectral regularization**:

$$\begin{aligned}
\frac{\partial}{\partial U} Obj_{rscs} &= \frac{\partial}{\partial U} \sum_{(\mathbf{x},\mathbf{z},\mathbf{y})\in C} \frac{1}{2} P_{\mathbf{x},\mathbf{z},\mathbf{y}} (\mathbf{x}-\mathbf{z})^T U^T \operatorname{diag}(V\mathbf{y}) U(\mathbf{x}-\mathbf{z}) \\
&= \sum_{(\mathbf{x},\mathbf{z},\mathbf{y})\in C} \frac{1}{2} P_{\mathbf{x},\mathbf{z},\mathbf{y}} \big(\operatorname{diag}(V\mathbf{y})^T U(\mathbf{x}-\mathbf{z})(\mathbf{x}-\mathbf{z})^T \\
&\qquad\qquad + \operatorname{diag}(V\mathbf{y}) U(\mathbf{x}-\mathbf{z})(\mathbf{x}-\mathbf{z})^T \big) \\
&= \sum_{(\mathbf{x},\mathbf{z},\mathbf{y})\in C} P_{\mathbf{x},\mathbf{z},\mathbf{y}} \operatorname{diag}(V\mathbf{y}) U(\mathbf{x}-\mathbf{z})(\mathbf{x}-\mathbf{z})^T \\
\frac{\partial}{\partial V} Obj_{rscs} &= \frac{\partial}{\partial V} \sum_{(\mathbf{x},\mathbf{z},\mathbf{y})\in C} \frac{1}{2} P_{\mathbf{x},\mathbf{z},\mathbf{y}} (\mathbf{x}-\mathbf{z})^T U^T \operatorname{diag}(V\mathbf{y}) U(\mathbf{x}-\mathbf{z}) \\
&= \frac{\partial}{\partial V} \sum_{(\mathbf{x},\mathbf{z},\mathbf{y})\in C} \frac{1}{2} P_{\mathbf{x},\mathbf{z},\mathbf{y}} (U(\mathbf{x}-\mathbf{z}) \circ U(\mathbf{x}-\mathbf{z}))^T V\mathbf{y} \\
&= \frac{1}{2} \sum_{(\mathbf{x},\mathbf{z},\mathbf{y})\in C} P_{\mathbf{x},\mathbf{z},\mathbf{y}} (U(\mathbf{x}-\mathbf{z}) \circ U(\mathbf{x}-\mathbf{z})) \mathbf{y}^T
\end{aligned}$$

Hence, for any choice of primary objective and one or more regularizers, we simply add the derivatives for each of $\mathbf{w}$, U, and V according to (4.7).

5.2 Second Trial

For the second online trial, we chose four algorithms again to randomly split between the LinkR application users. Social Matchbox was included again as a baseline since it was the best performing algorithm in the first trial. The four recommendation algorithms are:

- **Social Matchbox**: Matchbox MF + Social Regularization + L2 Regularization
- **Spectral Matchbox**: Matchbox MF + Social Spectral Regularization + L2 Regularization
- **Social Hybrid**: Hybrid + Social Regularization + L2 Regularization
- **Spectral Co-preference**: Matchbox MF + Social Co-preference Spectral Regularization + L2 Regularization

The online experiments were switched to the new algorithms on October 13, 2011. For the online results reported here, we took a snapshot of the data as it was on October 22, 2011.

5.2.1 Online Results

Results shown are the number like ratings and the number of dislike ratings normalized by the total number of ratings (likes + dislikes) per algorithm. Spectral Matchbox

Algorithm	**Users**
Social Matchbox	26
Spectral Matchbox	25
Spectral Co-preference	27
Social Hybrid	25

Table 5.1: Number of Users Assigned per Algorithm.

achieved the best ratio of likes to dislikes compared to the other algorithms. Like Social Matchbox in the first trial, Spectral Matchbox was the only algorithm to receive more likes than dislikes from its assigned users.

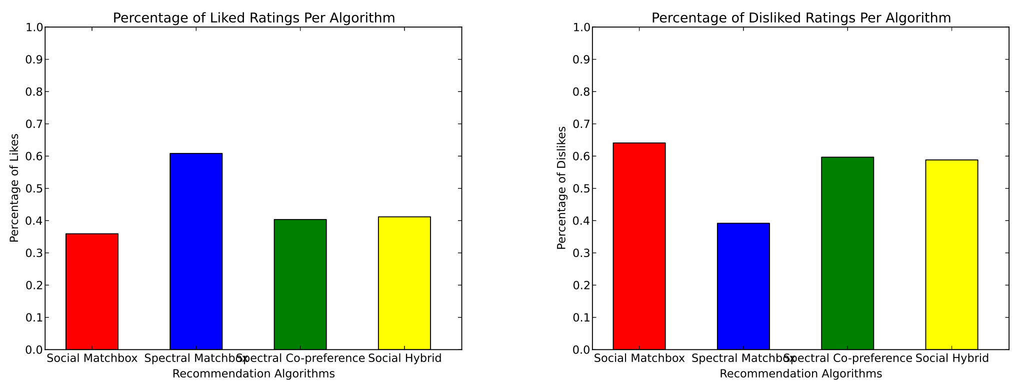


Figure 5.1: Results of online live trials. Spectral Matchbox achieved the highest ratio of likes to dislikes among the four algorithms.

We again split the results again between friend link recommendations and non-friend link recommendations, with the results shown being the number of likes or dislikes from friends or non-friends normalized by the total number of ratings on those type of recommendations. All four algorithms experiences a significant performance drop in the number of likes when it came to recommending non-friend links, which reflects the results of the first trial. Additionally, the differences were more drastic with the two algorithms that uses the social regularization method: Social Matchbox and Social Hybrid. This does seem to indeed show that aside from liking and disliking a link just from the quality of the links being recommended, users are also more likely to like a link simply because a friend had posted it and more likely to dislike it just because it came from a stranger.

5.3 Offline Results

When testing on all the combinations of the active data, the results between the algorithms on a test dataset are all within the standard error bars. The results of the live trials aren't reflected in the offline results, which highlights the difficulty of using the MAP metric for evaluating recommendation algorithms offline.

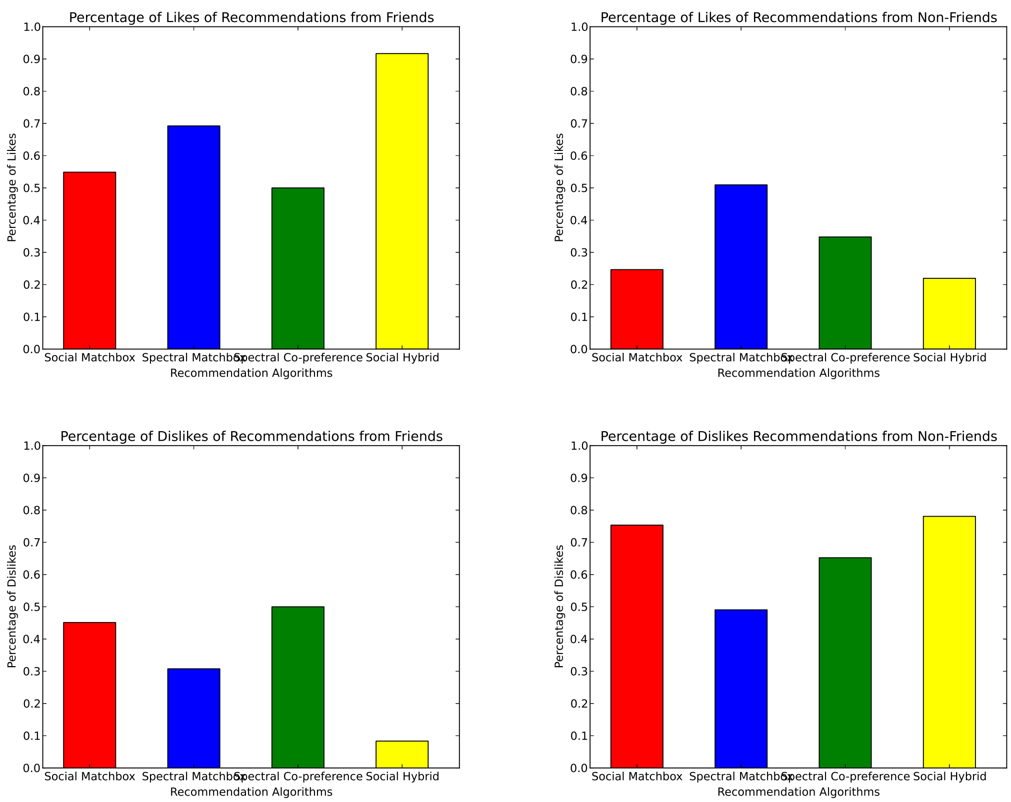


Figure 5.2: Results of the online live trials, split between friends and non-friends. As in the first trial, there is a significant drop in performance between recommending friend links and recommending non-friend links.

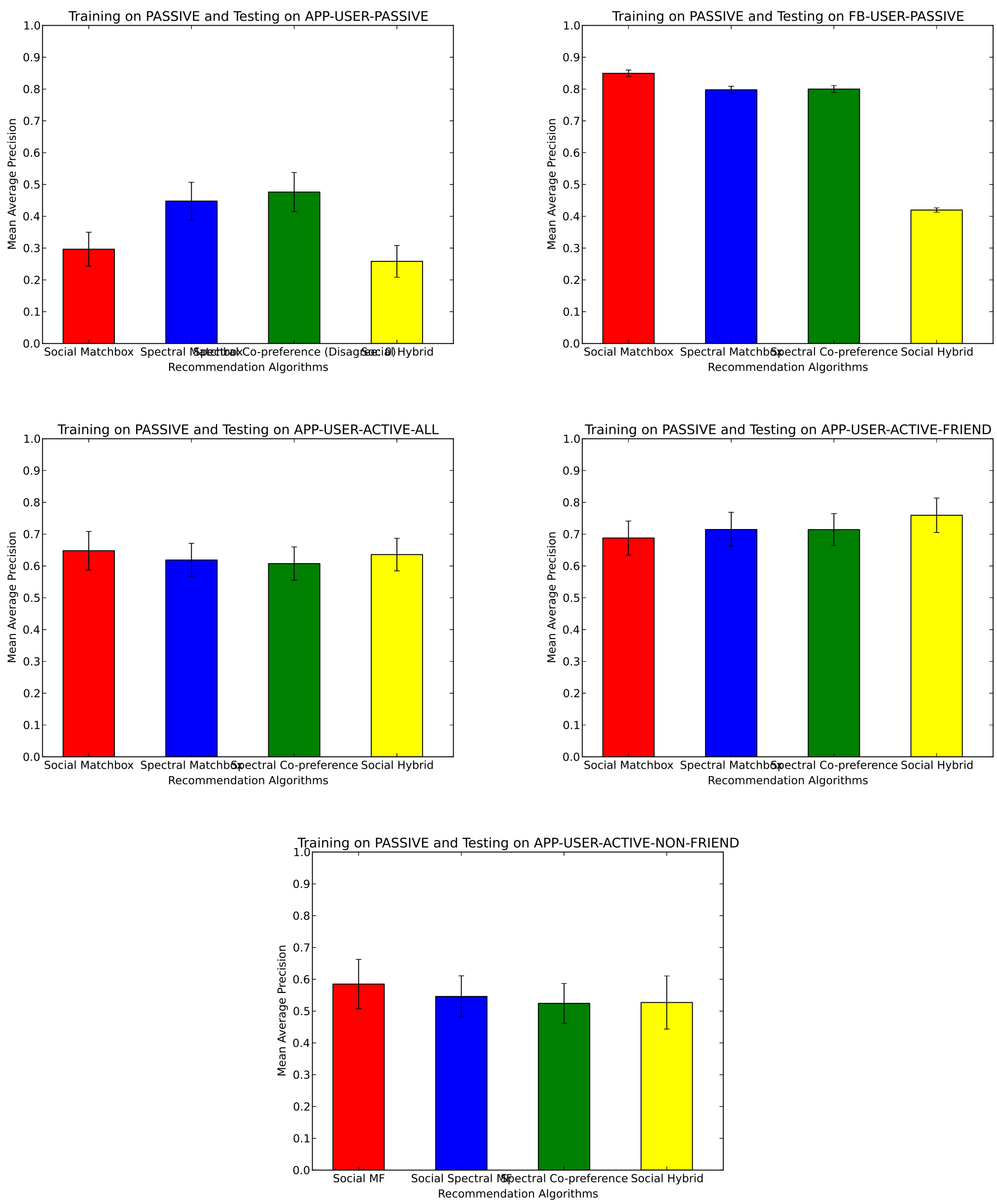


Figure 5.3: Results of training on Passive data

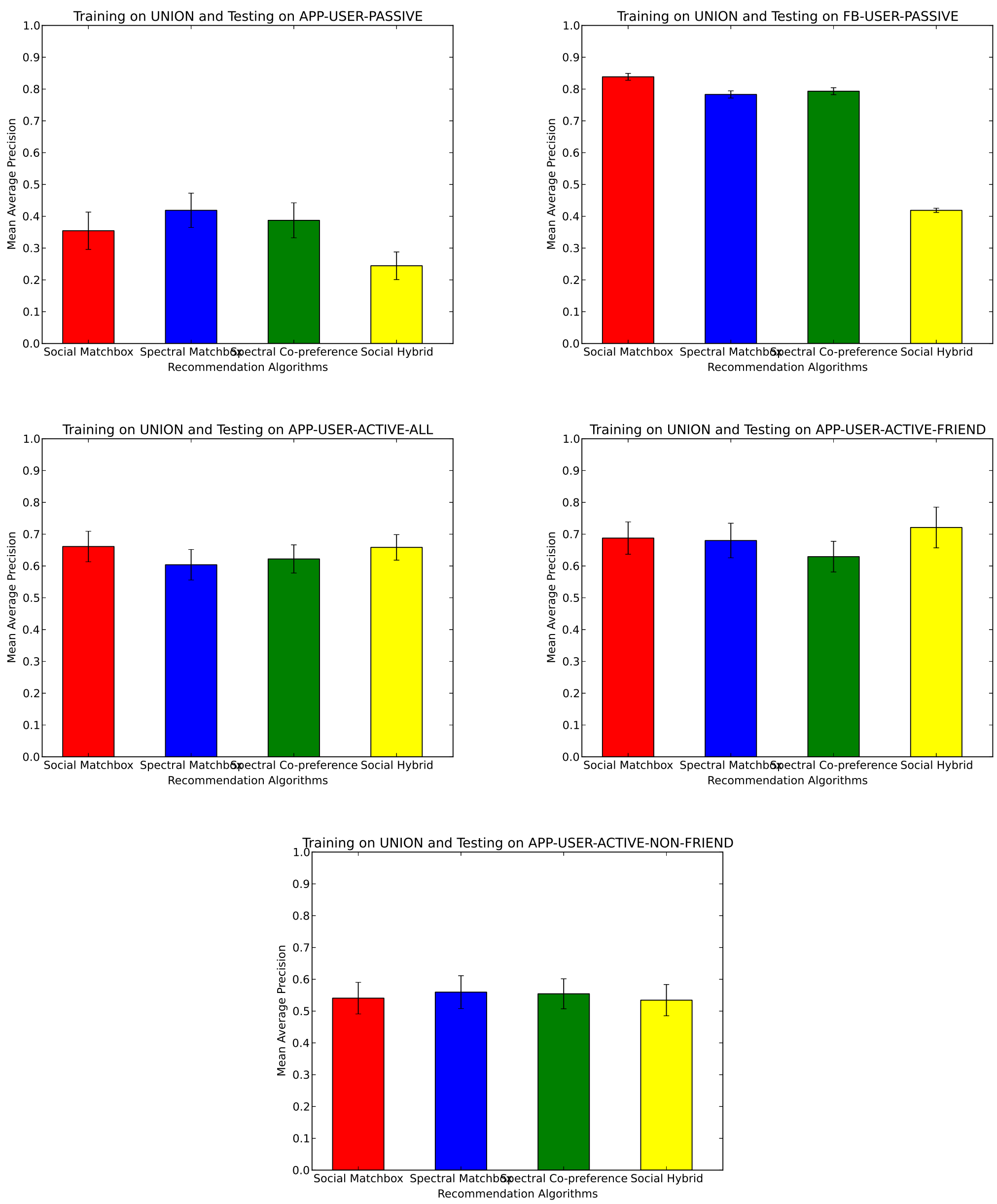


Figure 5.4: Results of training on Union data

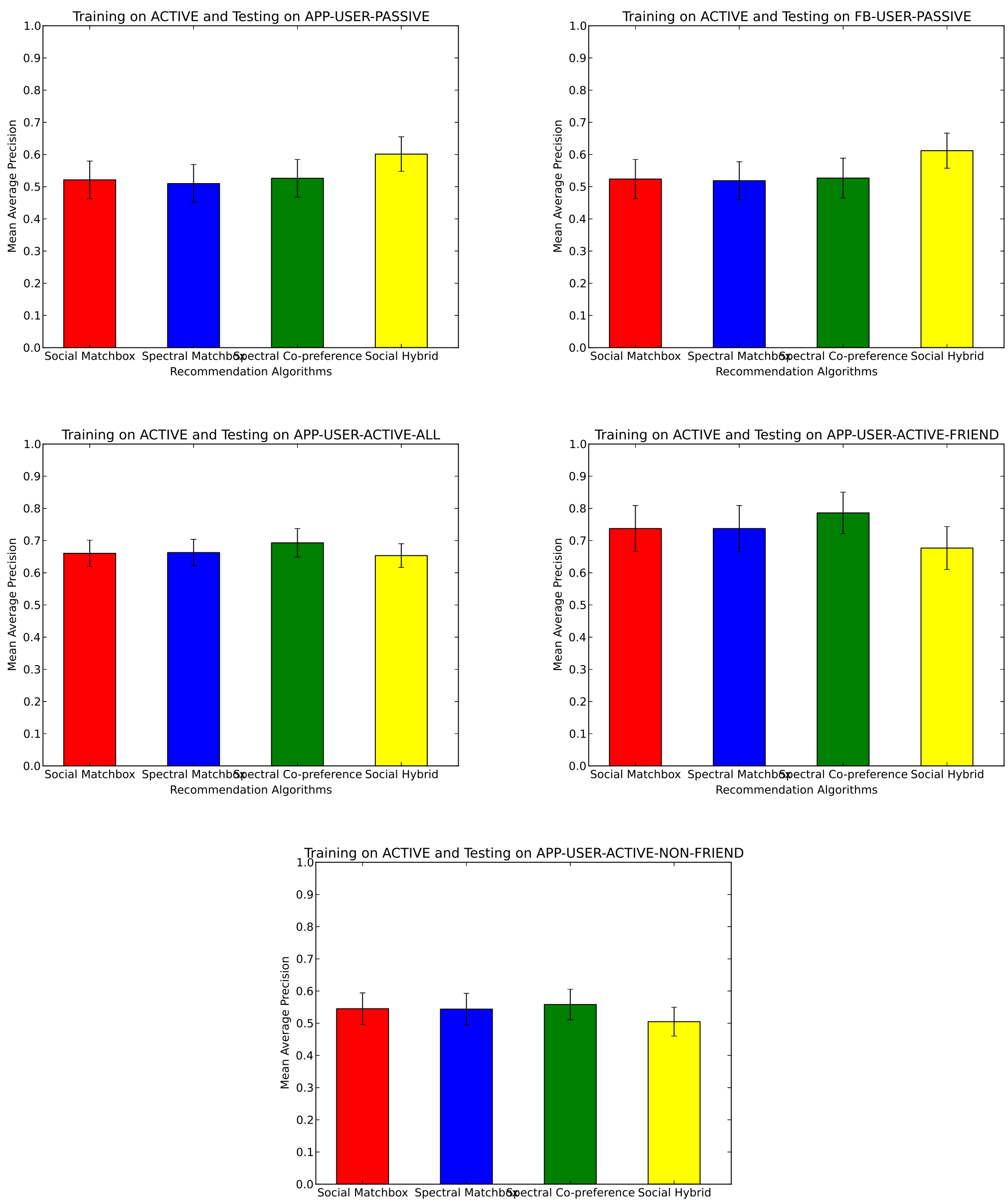


Figure 5.5: Results of training on Active data

Chapter 6

Conclusion

The Social Regularization constraint of Social Matchbox which has been detailed in other papers gives a very strong performance as a social recommender. It handily beat the other recommenders in the first trial and this was also reflected in offline experiments when training on the UNION dataset and testing on APP-USER-ACTIVE-ALL, the dataset combination that most closely resembles the training and recommendation data of the live trials. We also found that the Social Regularization constraint performed very differently when recommending friend links and when recommending non-friend links, and this was also reflected in the offline experiments.

The results of the second trial indicate that the Social Spectral Regularization constrain may be better for social regularization, the two algorithms that used it performed better than the two algorithms that used the Social Regularization constraint. However, this result can't be found in the offline experiments. What can be found in the offline experiments when training on the UNION data was same decrease in performance of the algorithms when recommending non-friend links compared to recommending friend links.

The results of the second trial highlight one difficulty encountered with evaluating the recommendation algorithms: which metric best correlates with human preferences. This paper uses the mean average precision as the main evaluation metric, but there may be other metrics that can better reflect the results of live user trials. Future work on social recommendation that may be crucial can be evaluation of different metric show closely they reflect user preferences.

The social recommenders discussed in this paper try to project the user preferences into the latent space, but it was found that other implementation issues greatly affect the user perceptions of the quality of the links being recommended. One complaint was that since the majority of the LinkR users were mainly English speakers, they automatically disliked links with non-English descriptions and those that pointed to non-English pages. A quick fix we did for this was to just stop recommending non-English links. In the future, having known languages in the user feature and the language of the link or its description in the link feature may result in a lot better and more accurate recommendation, especially for users knowledgeable in non-English languages.

With regards to link recommendations in LinkR, it was found that users were more likely to like recommended links when those links were originally posted by their friends, as opposed to links that were originally posted by non-friends. This was found in both the first and second user trials, and even in the offline experiments. This may be because users will trust a link more when it has been ”vouched” for by a friend posting it, rather than just any link from a random stranger.

Bibliography

Bell, R. M. and Koren, Y. 2007. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM-07* (2007). (p. 8)

Birlutiu, A., Groot, P., and Heskes, T. 2010. Multi-task preference learning with an application to hearing aid personalization. *Neurocomputing 73*, 7-9, 1177–1185.

Bonilla, E. V., Chai, K. M. A., and Williams, C. K. I. 2008. Multi-task gaussian process prediction. In J. Platt, D. Koller, Y. Singer, and S. Roweis Eds., *Advances in Neural Information Processing Systems 20*, pp. 153–160. Cambridge, MA: MIT Press.

Boutilier, C. 2002. A POMDP formulation of preference elicitation problems. In *Proceedings of the 18th National Conference on Artificial Intelligence* (Menlo Park, CA, USA, 2002), pp. 239–246. American Association for Artificial Intelligence.

Bradley, R. A. and Terry, M. E. 1952. Rank analysis of incomplete block designs: The method of paired comparison. *Biometrika 39*, 324–345.

Cao, B., Sun, J.-T., Wu, J., Yang, Q., and Chen, Z. 2008. Learning bidirectional similarity for collaborative filtering. In *ECML-08* (2008). (p. 8)

Chajewska, U. and Koller, D. 2000. Utilities as random variables: Density estimation and structure discovery. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence* (2000), pp. 63–71. Morgan Kaufmann Publishers Inc.

Chajewska, U., Koller, D., and Parr, R. 2000. Making rational decisions using adaptive utility elicitation. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence* (2000), pp. 363–369. AAAI Press / The MIT Press.

Chang, C.-C. and Lin, C.-J. 2001. *LIBSV: a Library for Support Vector Machines.* (p. 9)

Chu, W. and Ghahramani, Z. 2005a. Gaussian processes for ordinal regression. *Journal of Machine Learning Research 6*, 1019–1041.

Chu, W. and Ghahramani, Z. 2005b. Preference learning with Gaussian processes. In *Proceedings of the 22nd international conference on Machine learning* (New York, NY, USA, 2005), pp. 137–144. ACM.

Conitzer, V. 2009a. Eliciting single-peaked preferences using comparison queries. *Journal of Artificial Intelligence Research 35*, 161–191.

Conitzer, V. 2009b. Eliciting single-peaked preferences using comparison queries. *Journal of Artificial Intelligence Research 35*, 161–191.

Cui, P., Wang, F., Liu, S., Ou, M., and Yang, S. 2011. Who should share what? item-level social influence prediction for users and posts ranking. In *International ACM SIGIR Conference (SIGIR)* (2011). (pp. 8, 10, 20)

Eric, B., Freitas, N. D., and Ghosh, A. 2008. Active preference learning with discrete choice data. In J. Platt, D. Koller, Y. Singer, and S. Roweis Eds., *Advances in Neural Information Processing Systems 20*, pp. 409–416. Cambridge, MA: MIT Press.

Guo, S. and Sanner, S. 2010. Real-time multiattribute Bayesian preference elicitation with pairwise comparison queries. In *Thirteenth International Conference on Artificial Intelligence and Statistics* (2010).

Howard, R. 1966. Information value theory. *IEEE Transactions on Systems Science and Cybernetics 2*, 1, 22–26.

Huang, J. and Guestrin, C. 2009. Riffled independence for ranked data. In *In Advances in Neural Information Processing Systems (NIPS)* (Vancouver, Canada, December 2009).

Jones, D. R. 2001. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization 21*, 4, 345–383.

Kamishima, T. 2003. Nantonac collaborative filtering: recommendation based on order responses. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (New York, NY, USA, 2003), pp. 583–588. ACM.

Li, Yeung, and Zhang. 2011. Generalized latent factor models for social network analysis. In *IJCAI-11* (2011). (p. 8)

Li, W.-J. and Yeung, D.-Y. 2009. Relation regularized matrix factorization. In *IJCAI-09* (2009). (pp. 8, 10, 32)

Ma, King, and Lyu. 2009. Learning to recommend with social trust ensemble. In *SIGIR-09* (2009). (pp. 8, 10)

Ma, H., Yang, H., Lyu, M. R., and King, I. 2008. Sorec: Social recommendation using probabilistic matrix factorization. In *CIKM-08* (2008). (pp. 8, 10)

Ma, H., Zhou, D., Liu, C., Lyu, M. R., and King, I. 2011. Recommender systems with social regularization. In *WSDM-11* (2011). (pp. 8, 10, 32)

Quiñonero Candela, J. and Rasmussen, C. E. 2005. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research 6*, 1939–1959.

Rendle, S., Marinho, L. B., Nanopoulos, A., and Schmidt-Thieme, L. 2009. Learning optimal ranking with tensor factorization for tag recommendation. In *KDD-09* (2009). (pp. 8, 10)

Salakhutdinov, R. and Mnih, A. 2008. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, Volume 20 (2008). (p. 1)

Stern, D. H., Herbrich, R., and Graepel, T. 2009. Matchbox: large scale online bayesian recommendations. In *WWW-09* (2009), pp. 111–120. (pp. 3, 8, 19, 32)

Toscher, A. and Jahrer, M. 2009. The bigchaos solution to the netflix grand prize. (p. 1)

Viappiani, P. and Boutilier, C. 2009. Regret-based optimal recommendation sets in conversational recommender systems. In *Proceedings of the third ACM conference on Recommender systems* (New York City, NY, USA, October 2009), pp. 101–108. ACM.

Yang, Long, Smola, Sadagopan, Zheng, and Zha. 2011. Like like alike: Joint friendship and interest propagation in social networks. In *WWW-11* (2011). (pp. 8, 10, 20)

Yu, K., Tresp, V., and Schwaighofer, A. 2005. Learning Gaussian processes from multiple tasks. In *Proceedings of the 22nd international conference on Machine learning* (New York, NY, USA, 2005), pp. 1012–1019. ACM.