UNIVERSITY OF CALIFORNIA
SANTA CRUZ

TOWARD A SOCIAL GRAPH RECOMMENDATION ALGORITHM

A thesis submitted in partial satisfaction of the
requirements for the degree of

MASTER OF SCIENCE

in

ELECTRICAL ENGINEERING

by

Ali Adabi

June 2012

The Thesis of Ali Adabi
is approved:

Professor Luca de Alfaro, Chair

Professor James Davis

Professor John Vesecky

Tyrus Miller
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Acknowledgments

I consider myself very lucky to have the support and guidance of Luca de Alfaro on my thesis work. Luca motivated my research and development. I want to thank my friends and all of the volunteers who devoted their time to help me collect data for this work. I also would like to thank professor Davis and Vesecky for their constructive feedback and Diana Martinez for proof reading this thesis.

Chapter 1

Introduction

Web 2.0 is about joining communities, and connecting people to each other through social networks. Online content has been increasing drastically in recent years therefore making the desired information difficult to find. Recommendation systems strive to make this task easier by steering users to content relevant to their needs.

Recommender systems [2, 4] analyze previous user ratings,and preferences to offer the user personalized recommendations on items that may be of interest to them. Amazon or Netflix compare the purchases provided by users and recommend to each user new movies or products that have been watched or bought by users with similar purchasing habits. Pandora creates an interest graph for the users based on their musical preference to give personal recommendations.

Taste in movies is both sophisticated and personal. This makes movie recommendation very hard to predict. One might like *Usual Suspects* and dislike *Seven* even though both movies belong to the crime-thrillers. Therefore many algorithms focus more on user reviews (Collaborative Filtering) or dedicated tags (content-based) to get a prediction of what users might like.

In this thesis, we analyze the problem of movie recommendation and investigate ways to leverage the social graph in producing relevant recommendations for users. In the past approaches (*e.g.* Netflix and Pandora), the item recommendations are computed based on the taste similarity. We will explore making the recommendations also a function of what the friends of a user like or dislike. We believe this is particularly effective for movies since most people like to watch movies in company. We investigate the importance of movie recommendations through the user's social network. We believe showing the person the movies recommended and potential viewing companions may greatly enhance the appeal of our recommendations. We will explore different social recommendation algorithms for movies to prove the following two hypothesis:

- The knowledge of the social graph[1] of the user and the preference of their friends will enable us to make precise predictions. Since the taste of user in movies is hard to predict, having a contextual information about what

[1]Soical graph is the graph of relationships in a social network.

friends like enables us to weed out irrelevant recommendations. For example, a recommendation system based on Sam's preferences suggests three movies A, B, and C. If in this period the recommendation system has information about Sam's friends related to geographical location, age, and gender, it can weed out a non-related recommendations, such as B a foreign movie and C a movie favored by females.

- Moreover communicating to a user, not only the recommendation but also a list of potential friend with the same refinement enhances the likelihood of the recommendation's validity. Once validity is established between the user and the recommendation system, the user is more likely to trust the system.

Chapter 2

Related Work

2.1 Collaborative Filtering (CF)

One approach in designing recommendation systems is the collaborative filtering (CF) method. CF collects user's ratings and preferences in order to find similar users. CF methods can be divided into two categories: Neighborhood-based[1] and Model-based approaches. A large class of CF algorithms are Neighborhood-based [11].

Neighborhood-based CF algorithms use the entire or a sample of the user-item database to generate a prediction. A neighbor is defined as a user with similar taste. By identifying the neighbors of a new user, a prediction of preferences on

[1]Neighborhood-based algorithms are two types: User-based and Item-based. In this paper only the user-based method has been described.

new items for him or her can be produced [14].

The neighborhood-based CF algorithm, uses the following steps:

1. Calculates the similarity *sim(u,v)*, which reflects correlation between two user u and v (2.1).

2. Produces a prediction value($p_{u,i}$) (2.3) for the active user by taking the weighted average of all the ratings of the user on a certain item (2.3).

Several approaches have been used to compute similarity between users. The two most common approaches are *correlation-based* and *cosine-based*.

The most commonly correlated-based measure of similarity is the *Pearson* correlation coefficient (2.3) between the ratings of the two users. *Pearson* correlation measures the extent to which two variables linearly relate to each other [9]. The Pearson correlation between users u and v is given by equation (2.1) where $r_{u,i}$ is the rating of user u on item i. $i \in I$ summations are over the items that both the users u and v have rated and $\bar{r}_u$[2] is the average rating of the co-rated items of the uth user:

$$sim(u,v) = \frac{\sum_{i \in I}(r_{u,i} - \bar{r_u})(r_{v,i} - \bar{r_v})}{\sqrt{\sum_{i \in I}(r_{u,i} - \bar{r_u})^2}\sqrt{\sum_{i \in I}(r_{v,i} - \bar{r_v})^2}} \tag{2.1}$$

In the *cosine-based* (2.2) approach the two users u and v are treated as two vectors

[2]Note $\bar{r}_v$ is the average rating of the co-rated items of the vth user.

in n-dimensional space.

$$sim(u,v) = \frac{\sum_{i \in I} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in I} r_{u,i}^2} \sqrt{\sum_{i \in I} r_{v,i}^2}} \tag{2.2}$$

where $r_{u,i}$ is the rating of user u for item i and $r_{v,i}$ is the rating of user v for item i.

To obtain predictions or recommendations is the most important step in a collaborative filtering system. In the neighborhood-based CF algorithm, a subset of nearest neighbors of a user are chosen based on their similarity with him or her. A weighted aggregate of their ratings is used to generate predictions for the user. To make a prediction for the user u on a certain item i, we take a weighted average of all the ratings on that item according to the following formula [12, 14] :

$$p_{u,i} = \bar{r}_u + \frac{\sum_{v \in U} sim(u,v).(r_{v,i} - \bar{r}_v)}{\sum_{v \in U} sim(u,v)} \tag{2.3}$$

where $\bar{r}_u$ and $\bar{r}_v$ are the average ratings for the user u and user v on all other rated items, and $sim(u,v)$ is the similarity between the user u and user v. The summations are over all the users $V \in U$ who have rated the item i.

2.1.1 Collaborative Filtering Challenges

Collaborative filtering faces several challenges:

- **Efficiency**

  Calculating *predictive values* ($P_{u,i}$) is expensive requiring calculating *sim(u,v)*,

the similarity of u against all of the other users $v \in U$. Therefore a subset of users should be selected before computation, and clustering can be used to locate a *similar user* (neighbor).

- **Cold Start**

  The CF systems need considerable amount of existing information on the user's past preference history in order to make an appropriate decision. This is due to the fact that CF accumulates other users' ratings to make a decision for a target user.

- **Scalability**

  Nearest neighbor algorithms require computation that grows with both the number of users and the number of items. With millions of users and items, a typical web-based recommender system running existing algorithms will have serious scalability problems [13].

- **Sparsity**

  The sparsity problem occurs when available data is insufficient for identifying similar users (neighbors) therefore create a major issue that limits CF [8].

Table 2.1 shows a series of user preferences along with the items each user has rated. The cells with values "?" correspond to items for which users did not vote on. CF calculates values of "?" cells by looking at similarities between items

that each user voted on.

Table 2.1: Shows an example of users' movie tastes.

	Alice	Bob	John
Up	Like	?	Like
Shrek	Like	Dislike	Like
Batman	Dislike	Like	?

For Instance, *Alice* and *John* liked the movies *Up* and *Shrek* while Alice dislike *Batman* in table 2.1. Therefore CF does not recommend *Batman* to *John* since *John*'s taste is similar to *Alice*.

2.2 Content-Based Filtering(CBF)

User profiles contain compact descriptions of users' interests and personal preferences. Content-based Filtering represent each item (e.g. movie, song, etc.) as a vector of information about the item. Looking at the user's interest, CBF recommends items based on user to item affinity and item to item similarity [10].

Content-Based Filtering Algorithm is described below:

1. Quantify a series of characteristics in items.

2. Generate predictions based on similarities between items and recommend them to a user with similar items.

In Table 2.1, CBF concludes that Bob dislikes the movie Up. Since movies Up and $Shrek$ are similar and Bob dislikes $Shrek$ CBF predicts he dislikes the movie Up. Therefore CBF does not recommend Up to Bob. Equation (2.4) describes how the prediction score $p(u, i)$ for user u and item i is computed:

$$p(u, i) = \sum_{k=1}^{m} sim(i, k).v_{u,k} \tag{2.4}$$

where $sim(i, k)$ is the measure of similarity between items k and i, $v_{u,k}$ is the vote of user u for item k. We consider all m items rated by user i. Therefore to compute how much user u likes item i, we consider all items k that user u has rated, then we sum the similarity between k and our target item i and multiplied it by the rating received by k.

2.3 CF vs. CBF

Any recommendation scheme has pros and cons. For instance, CF and CBF both suffer from the cold start problem. However compared to the CBF approach, according to *Burke et. al.* [4] CF can recommend cross-genre or 'outside the box'. It may be that people who enjoy crime-thrillers also enjoy comedy, but a CBF recommender trained on the preferences of a crime-thrillers would not be able to suggest items in the comedy category since none of the features (Actors, Script, Genre, etc.) associated with items in the different categories are shared. Only by

looking outside the preferences of the individual can such suggestions be made.

The greatest strength of CF is that it is completely independent of any complex object (e.g. content) being recommended. Therefore CF is suitable for complex objects such as music and movies where variations in taste are responsible for much of the variation in preferences. Table 2.2 shows a comparison between different recommendation algorithms.

CBF solves the sparsity problem mentioned 2.1.1. Therefore, even if the matrix is sparse as long as a user has a series of preferences CBF can use this information to generate predictions.

Table 2.2: Recommender techniques [4]

Technique	**Background**	**Input**	**Process**
Collaborative	Ratings from U of items in I.	Ratings from u of items in I.	Identify users similar to u, and extrapolate from their ratings of i.
Content based	Features of items in I.	u's ratings of items in I	Generate a classifier that fits u's rating behavior and use it on i.

2.4 Enhancing Collaborative Filtering with Friends

Today's social networking websites such as Facebook, LinkedIn, and Twitter are a hub for many users to share interest and other information with their friends or people of interest. Such social relationship can be very effective for generating recommendation compared to traditional Collaborative Filtering methods. Using

the FilmTrust system as a foundation, *Golbeck* [7] showed that these recommendations are more accurate than other techniques when the user's opinions about a film are divergent from the average.

Recommendations from people the user is *familiar* with could be more *relevant* to the user. Users can explicitly engage with their friend therefore making recommendations more engaging and effective. Various works [8, 7, 3] have shown the effectiveness of friend based recommendation over Collaborative Filtering.

2.5 Social Recommender Systems

Social Recommender Systems (SRSs) target the social media domain. The goal of SRSs is to improve quality of recommendation and solve the problem of social information overload. Recommendations based on friends have advantages of familiarity, similarity, and trust. Categories of SRS systems are shown in the table 2.3.

SRS needs to have background information on individuals, such as movies they liked in the past. This background information help SRS define individual's taste in movies. Good explanations inspire trust and loyalty and increase user satisfaction.

Table 2.3: SRS category with a few examples [5]

Targets	**Examples**
People	ReferralWeb (Kautz et al., 1997), Expertise Recommender (McDonald and Acerman, 2000), SonarBuddies (Guy et. al 2008) Do You Know (DYK)(Guy et. al 2009b)
Multimedia	Flickr, Youtube, Flixter, FilmTrust (Golbeck and Hendler 2006)
Cool Content	Pinterest, Instagram
Groups	Facebooks Fan Page, LinkedIn Group
Food recipes	Kalas (Svensson et. al 2005)
Ski tracks	Moleskiing

Chapter 3

Recommendation via Friends

Social networking websites such as Facebook have became a prominent source of information sharing recently. People are willing to share their interests on social networking sites. On Facebook (901 million users as of March 2012), users are share than 3.2 billion likes per day. [6] facebook API provides the information needed for social recommendation engines. In this work different experiments have been done with the purpose of recommending users a set of movies users will find relevant. Figure 3.1 shows the relationship of user to the friends network movies.

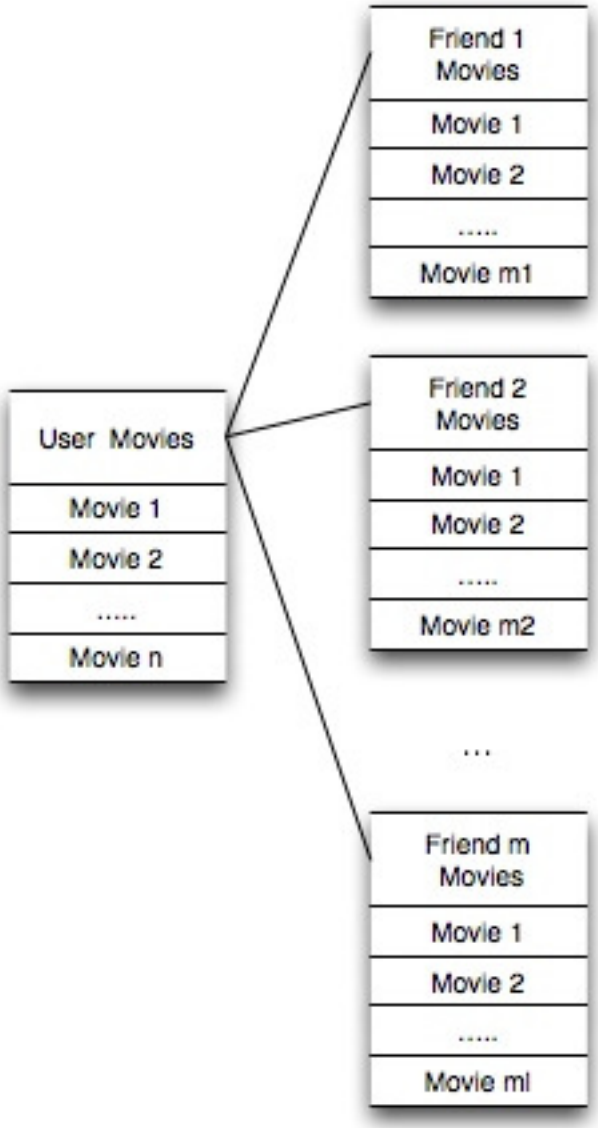

Figure 3.1: User friend network movies

3.1 Friends Network Algorithms

3.1.1 Basic Social Recommendation (BSR):

BSR finds the most frequent recommended movies by friends. It works as follows:

Algorithm 1 Basic Social Recommendation (BSR):

1: **for all** $u \in users$ **do**

2: **for** every u's friend v of u **do**

3: Extract the list of v's movie likes *(recommendations)*

4: Count the number of *movies* v liked

5: **end for**

6: Sort the movies list based on like counts in line 4

7: **return** Top movies calculated in line 6

8: **end for**

BSR scans all movies users friends liked, and generates a list of highly recommended movies among each users social network.

3.1.2 General Stranger Recommendation (GSR):

The sample participants in the experiment preformed in chapter 4 is small(20 participants). Therefore to throughly understand the effect of social graph on the user's decision, we introduce the GSR algorithm which scans interests of the user

non-friends(v') or strangers. Therefore if $v' \notin u(v)$ statement mean friend v is not user u's friend. The GSR algorithm goes through u's (user's) all v' (strangers) in the database to recommend items. GSR works as follows:

Algorithm 2 General Stranger Recommendation (GSR):

1: **for all** $u \in users$ **do**

2: **for** every v' **do**

3: Extract the list of v' movie likes *(recommendations)*

4: Count the number of *movies* v' liked

5: **end for**

6: Sort the movies list based on like counts in line 4

7: **return** Top movies calculated in line 6

8: **end for**

3.1.3 Explanation Social Recommendation (ESR):

Many recommender systems are providing no transparency into the working of the recommendation. Explanations provide that transparency, exposing the reasoning and data behind a recommendation. The ESR algorithm explains to the user where the social recommendation results are coming from. ESR displays friends faces and their name next to the recommendations it provides. Therefore, it provides the source of where the recommendation is coming from making the

recommendation results more transparent.

Algorithm 3 Explanation Based Recommendation (ESR)using BSR :

1: **for all** $u \in users$ **do**

2: **for** every u's friend v of u **do**

3: Extract the list of v's movie likes *(recommendations)*

4: Let f_m be Count the number of *movies* v liked

5: **end for**

6: Sort the movies based on f_m values calculated in line 4

7: **return** Present top movies in line 6, showing identification of recommenders (picture and name)

8: **end for**

3.1.4 Clustering Social Recommendation (CSR):

CSR is a clustering method based on user's social graph. This algorithm (4) uses the social graph of a user to find friends with similar taste in movies. If v is the user u's similar friend, CSR recommends the remaining movies that v has watched and u has not. CSR uses equation (3.1) to calculate $\bar{P}(u_i, v_j)$ the percentage similarity between user i (u_i) and friend j (v_j).

$$\bar{p}_i = \bar{P}(u_i, v_j) = \frac{1}{m} \sum_{k=1}^{n} \sum_{p=1}^{m} sim(um_k, vm_p) \tag{3.1}$$

where $sim(um_k, vm_p)$ is the cosine similarity between the movie pairs um_k and

vm_p given by equation (2.2). Cosine similarity fluctuates between 0 (no match) and 1 (perfect match), but for simplicity of this experiment, here the similarity is either 0 for no match or 1 for a perfect match(Exact Match). CSR calculates $\bar{p}_i$ for all friends (v_j) and sorts v_j based on it's associated similarity percentage $\bar{p}_i$. CSR then recommends to u_i, the movies with higher percentage similarity $\bar{p}_i$ which are not in user v_j's movie list.

Algorithm 4 Clustering Social Recommendation (CSR):

1: **for all** $u \in users$ **do**

2: **for** every user u's friend as v_j **do**

3: Extract the list of v_j's movie likes *(recommendations)*

4: let $\bar{p}_i = \bar{P}(u_i, v_j) = \frac{1}{m} \sum_{k=1}^{n} \sum_{p=1}^{m} sim(um_k, vm_p)$

5: **end for**

6: Sort friend list v based on each v_j percentage similarity $\bar{P}(u_i, v_j)$

7: Let R_m (Remaining movies) be the list of Movies v_j likes and u_i don't

8: **return** R_m based on their rank in line 6

9: **end for**

3.1.5 Clustering Based Recommendation (CBR):

CBR is a generic form of CSR. Instead of going through user friends, CBR goes through all strangers (not-user-friends) as algorithm 5 describes. The CBR

serves as a base case for CSR in identifying the benefits or shortcomings of social network recommendations. CBR traverses all of the social network data to find a

Algorithm 5 Clustering Based Recommendation (CBR):

1: **for all** $u \in users$ **do**

2: **for** every user u's non-friend as v'_j **do**

3: Extract the list of v'_j's movie likes *(recommendations)*

4: let $\bar{p}_i = \bar{P}(u_i, v'_j) = \frac{1}{m} \sum_{k=1}^{n} \sum_{p=1}^{m} sim(um_k, v'm_p)$

5: **end for**

6: Sort friend list v' based on each v'_j percentage similarity $\bar{P}(u_i, v'_j)$

7: Let R_m (Remaining movies) be the list of Movies v'_j likes and u_i don't

8: **return** R_m based on their rank in line 6

9: **end for**

similar user as u_i. In traversing, CBR discards all of u_i's friends.

Chapter 4

Experiments on the Effectiveness of Social Algorithms: BSR, GSR, ESR, CSR, and CBR

A series of experiments in this chapter (4) have been implemented on the social algorithms BSR, GSR, ESR, CSR, and CBR which have been described in chapter 3. Sample size were 20 college students ages between 20 to 28. The size of the movie database in this study was 5000 titles. Users were asked to login to a facebook application using their credentials. Upon successful authentication, facebook application was able to extract user and his or her friends' movie information. Users were shown a series of recommendations (BSR, GSR, etc.) and

their answers were recorded.

4.1 BSR vs. GSR

In this experiment, 5 GSR recommended movies were combined randomly with 5 BSR movies and have been offered to the users. Users were asked to pick if they like or dislike any of the 10 recommendations provided(so the results are not skewed). The displayed results gave users equal chances of choosing BSR or GSR movies. This study tested if users liked BSR or GSR movies without prior knowledge of their friends movie preferences.

As figure 4.1 shows majority of users (83%) liked either BSR or GSR movies(about the same) with BSR edging GSR. Its notable to consider that 17% of the recommended movies were disliked by the users (or users found them irrelevant). The reason that the values of BSR and GSR in figure 4.1 are similar could be due to the fact that the user sample is uniformly chosen from college students (age 20-28) and perhaps a non-uniform sample could change the results toward the BSR recommendations.

The overall BSR vs. GSR results showed that since user's friends are generally similar in terms of age, social class, and geographical location, an overlap in interests is visible. Although these interests are not uniform since friends are not uniform, but the integration of all friends' interest can be generally used as the

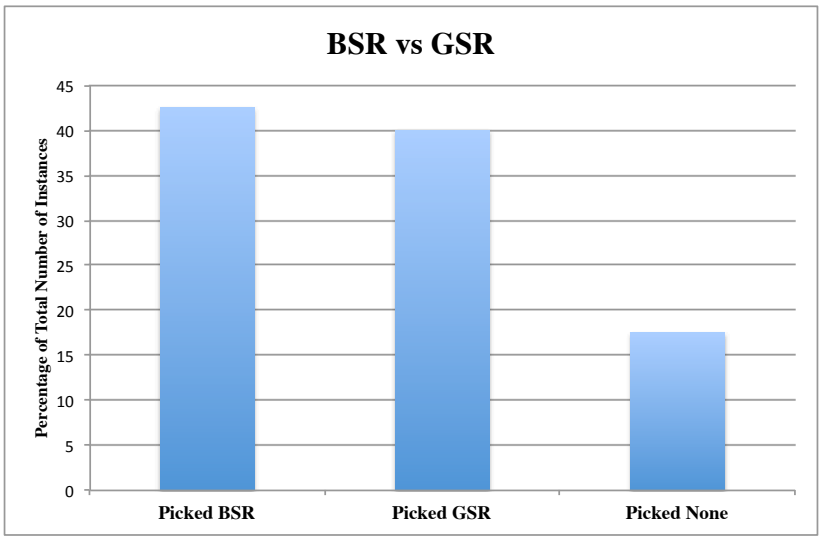

Figure 4.1: BSR vs. GSR

user's preference as well.

4.1.1 Gender Attribute

Friends' gender is also a defining factor in choosing a movie title. Knowing that generally male and female users like masculine and feminine movies respectively can provide a filter for the results mentioned above. Applying a gender classification filter to the recommendations can lead to more personalized results. The gender filter was applied to the recommended movies which eliminated opposite sex recommendation and resulted figure 4.2.

Figure 4.2 shows adding a gender classifier can enhance the recommendations performance however anomalies can also be created. One of the visible issue

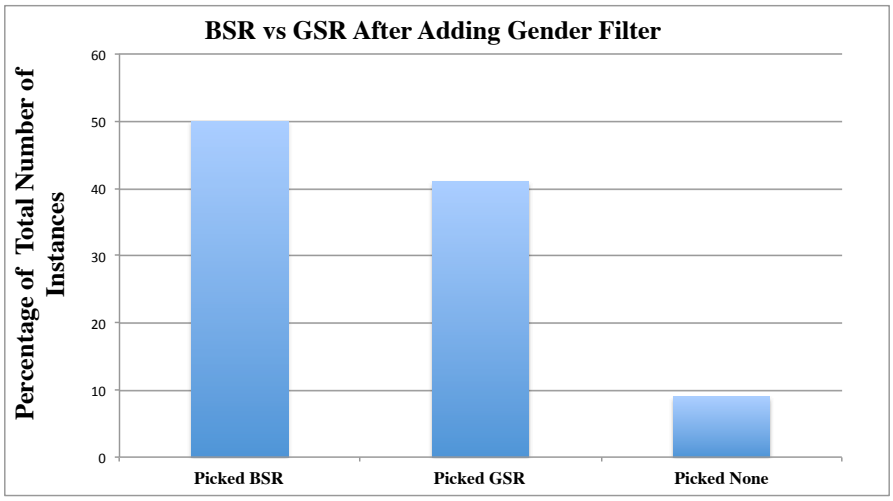

Figure 4.2: BSR vs. GSR after applying a gender filter (Number of total instances was 200)

was elimination of some independent and not mainstream movies, which have been discarded, in the recommended movies specifically in the male participants results.

4.2 CSR vs. CBR

CSR finds people with similar taste within the friends' network. The experimentation was conducted to compare the effectiveness of recommendations of similar users between two distinct groups: Friends of a users(CSR) and Strangers(CBR). similar strangers. Figure 4.3 shows the CSR and CBR recommendation comparison among users.

Users preferred CSR results 21% more than CBR. This is because if a user u

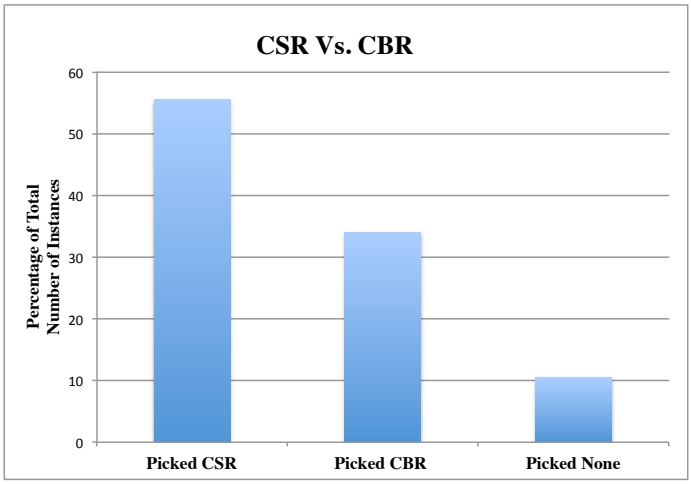

Figure 4.3: CSR vs. CBR for 200 instances

has a friend v, u already has some similarities to v. These similarities may be cultural, viewpoints, geological location or economical class of v. On the other hand searching the global network for similar users with CBR resulted in fairly good out come. In our experiment the sample users are small(20). However, an increase in number of users can increase chances of finding a more similar match.

4.3 ESR

In a separate experiment, ESR algorithm was compared to the BSR. ESR is similar to the BSR, but different because it shows the recommenders' faces along with the results. This experiment measures if explicit using of friends recommendations has an affect on user decision by showing friends faces next to the movie

recommendation. We measured the of effect ESR on user u's decisions in cases A and B. In the Case A, we studied effect of identity of friend v's on u's decision, and in Case B, we varied the number of v's profile pictures(face count) to study the effect.

4.3.1 Case A: Effect of Individual Friends

In this case, individual user friends' explicit effect on the user decision were examined. Users could see the recommended movies as before, except the recommenders' faces were displayed next to the movies. Limiting recommenders' face count to 5, allowed users to judge the results mainly by looking at the recommender's face. Each user expressed his or her opinion on the movies. Users stated their intention on watching or not watching movies after observing their friends faces who recommended them.

Figure 4.4 clearly illustrates showing faces made users decide on wishing to watch or not wishing to watch a movie. Past experiences helped users make their decisions when the faces were recognized and their faces was associated with good ore bad taste. Participants valued a group of friends they recognized as individuals with great taste or tastemakers. When users saw the selected group of tastemakers recommended a specific movie, they instantly wanted to follow other movies they suggested.

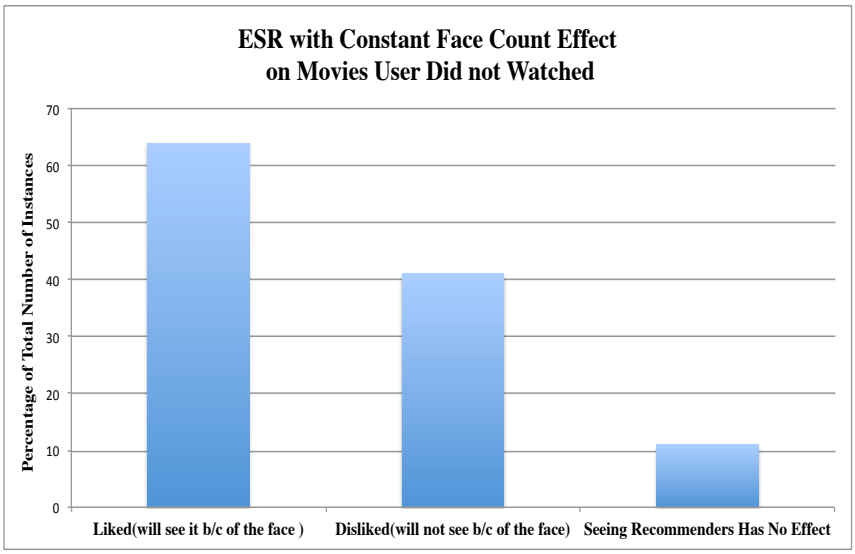

Figure 4.4: ESR with constant face count effect calculated for 200 intances

4.3.2 Case B: Increasing number of recommenders' faces

In Case B, we varied the number of friends displayed next to the recommended movies from 1 to 10. We asked users if they would like to watch a movie they have not watched because a number of their friends recommended it.

As figure 4.5 shows recommender numbers has a direct effect on the user decision. This could be because the majority of the movies that have been liked by users, are in fact popular movies among users social network and therefore are relevant to the users. In recommendations with lower number of recommenders (ex. for 3 recommenders), when some of the users saw a specific friend watched the movie they disliked it. This is in par with the assumption made in Case A.

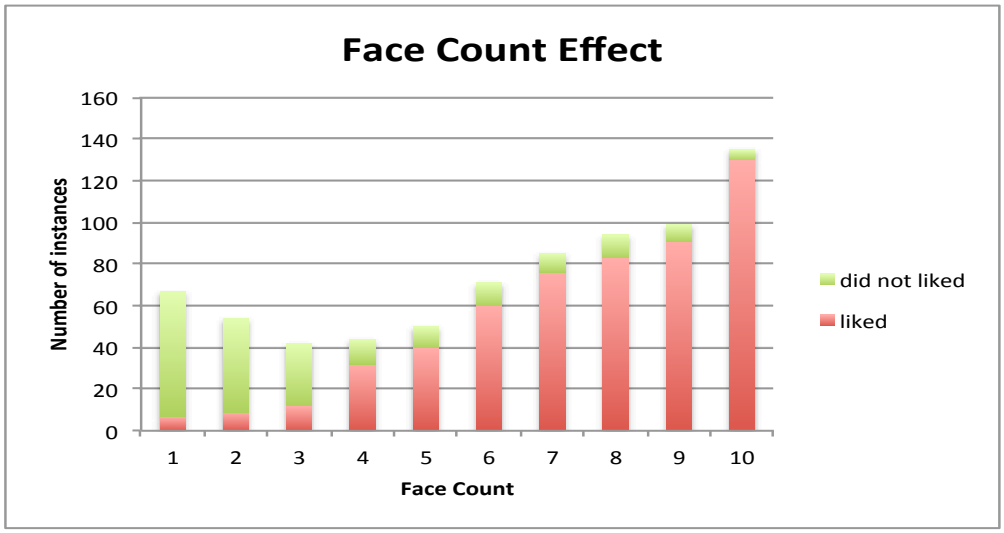

Figure 4.5: face density effect

However once the face count of the recommenders increase and pass a threshold, users tend to prefer the movies with higher count of friends faces.

4.4 *ESR* vs. *BSR*

In this case each user is given 10 BSR movie recommendations followed by showing their recommender faces (ESR) for the movies user have not yet watched. Users were asked if after knowing the movies their friends had recommended, they would like to change their decision. For instance, We showed Sam the movie *Up* provided by *BSR*. Sam decided to watch it, then we showed his friend's faces John and Mike who recommended the movie. Then Sam was asked if knowing

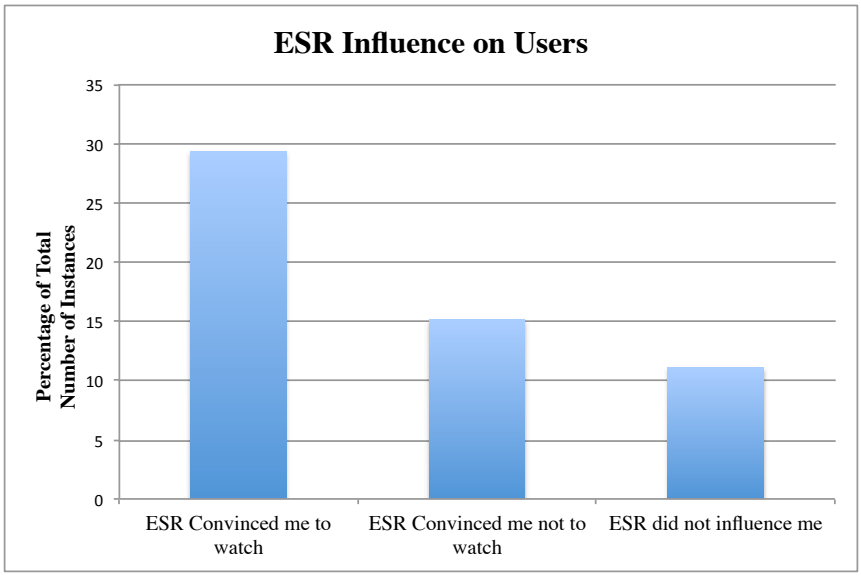

Figure 4.6: ESR influence for 200 instances

the fact that John and Mike recommended the movie, changed his mind (4.6).

As figure 4.6 shows 80% of users are convinced to make a decision if they want or not want to watch a movie after ESR. This means the explanation based recommendation has an important effect on users decision. However as figure 4.6 illustrates, surprisingly in 26% of total of instances (times of movies picked), users decided not watch a movie after knowing some of their friends watched a movie. One in every three individuals decided not to watch a movie because their friend watched movie. This exposes a new side of social recommendation, which is their negative effect on users decision. This negative effect is more significant for a certain threshold of movie popularity. When the level of movie popularity is

mediocre, and the user is doubtful, user has one in three chances of deciding not to watch the movie because of the friends' identity.

4.5 Results Analysis

One of the problems with the current methodology of collaborative filtering is that when a new movie comes out, there is not sufficient rating data about the movie. A particular movie that a friend likes has a higher chance of being recognized by similar users. Abdul-Rahman and Hailes [1] showed that in a predefined context, such as movies, users develop social connections with people who have similar preferences. Ziegler and Lausen[16] extended these results in work that showed a correlation exists between trust and user similarity in an empirical study of a real online community. Clearly, recommendations only make sense when obtained from like-minded people [16].

Social recommendations used in the studies above show that the overall advantage of this method is relevancy of the result and trust that user have established with the recommender through their social network. As *Papagelis et al.* [8] states, retrieving data from the user and his social graph intrinsically alleviates the sparsity problem. However, the algorithms used to analyze friends is memory and resource consuming. BSR for example takes in the order of $O(N \times M)$ per user to analyze where N is the number of user's movies and M is the number of

friends' movies. For CSR the timing complexly of the system is in the order of $O(P \times N \times M)$ where P is the total $users$. In summary social movie analysis are facing challenges regarding Insufficiency of data, Cold Start and Runtime:

- Insufficiency of data: Some users friend might not declare their interests in their profile while some might overly exaggerate about their interest creating anomalies.

- Cold Start: Users might not initially like a lot of movies this problem can be resolved if the user inputs movies they liked into their facebook profile.

- Run time: Timing complexity order of $O(P \times N \times M)$ may be a problem considering a large database. One common strategy for reducing the overhead to the system is to pre-calculate all user similarities $sim(u_i, v_k)$ in advance and recalculate them only once in a while (since the network of peers usually does not change dramatically in a short time). Then, whenever the user asks for a recommendation, the ratings can be efficiently calculated on demand using pre-computed similarities[2] .

However social recommendations are extremely important and 80% of the users change their mind about which movie they want to see after seeing them. We also found a very surprising negative effect about social recommendations that sometimes they are useful not for the movies you want to watch, but for the

movies that you don't want to be watching.

Chapter 5

Conclusion

In this thesis, social recommendation and it's impact on user's decision was studied looking at various different methods. It appears that social recommendation is effective in helping users decide in 80% of instances. This could be due to the fact that users trust or distrust some of their friends. When the count of recommenders is less than 3, 1 out of 2 users mentioned they *will not see* a specific movie because a *non-similar* friend they knew has watched a movie. This exposes a different side of the recommendation algorithms that has not been explored before.

Explanation-based social recommendation(ESR) produced a surprising side of social recommendations and that is their negative effect on users decision. One in every three individuals decided not to watch a movie because their friend has

watched movie. This negative effect is more significant in movies with mediocre level of popularity. The mediocre level of popularity in movies conveys doubt in users's decision. This gives users a '50-50' chance of getting swayed one way or the other. *One* in *three* users decided *not* to *watch* a *mediocre* movie(with mediocre level of popularity) because of their *friends*' identity.

Social recommendations offered a new type of information that can be used separately or as a hybrid method for the current known methods such as collaborating filtering or content-based methods mentioned in the introduction.

5.1 Future Work:

A hybrid method should be formed integrating the current method of social recommendation to a content-based approach. A content-based method should associate every movie with a vector containing the essence of every movie. When finding similarity between movies, a content-based approach can identify the categories of movies and find out how similar the movies are. Therefore when calculating $\bar{p}$ similarity $sim(um_k, vm_p)$ can be the cosine similarity of two movie vectors (um_k and vm_p correspond to the content of the movie as (3.1) described).

Further, a learning algorithm should be developed for identifying the movie tastemakers among friends. Tastemakers have a huge effect on decision of users. They can expose users to unique set of quality and diverse recommendations.

Sometimes finding tastemakers among friends is not easy since they might not share much information. The users normally favored the quality of the tastemakers' recommendations even though they are small in quantity.

Bibliography

[1] A. Abdul-Rahman and S Hailes. Supporting trust in virtual communities. *33rd Hawaii International Conference on System Sciences*, 2000.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Towards the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):634–749, 2005.

[3] Maurice Coyle Barry Smyth, Peter Briggs and Michael OMahony. Google shared.a case-study in social search. *17th International Conference, UMAP*, 2009.

[4] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, November 2002.

[5] Yu Chen. Social Recommender Systemsmethods and user issues. `http:`

//hci.epfl.ch/teaching/advanced-hci/slides/2011.5.23_Yu.pdf. Accessed: 5/8/2012.

[6] facebook. Key Factsstatistics. http://newsroom.fb.com/content/default.aspx?NewsAreaId=22. Accessed: 5/20/2012.

[7] Jennifer Golbeck. Generating predictive movie recommendations from trust in social networks. *iTrust*, 2006.

[8] Themistoklis Kutsuras Manos Papagelis, Dimitris Plexousakis. Alleviating the sparsity problem of collaborative filtering using trust inferences. *iTrust*, 2005.

[9] M. Suchak P. Bergstrom P. Resnick, N. Iacovou and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. *ACM Conference on Computer Supported Cooperative Work*, 1994.

[10] Michael J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artifficial Intelligence Review*, pages 393–408, 1999.

[11] Vikas Sindhwani Prem Melville. *Recommender Systems*. Encyclopedia of Machine Learning, New York, NY, 2010.

[12] F. Ricci, Rokach, and L. *Recommender Systems Handbook*. Springer, New York, NY, 2011.

[13] B.M. Sarwar, G. Karypis, J.A. Konstan, and J. Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. *The Fifth International Conference on Computer and Information Technology (ICCIT 2002)*, 2002.

[14] Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, August 2009.

[15] Hsinchun Chen Zan Huang, Daniel Zeng. A link analysis approach to recommendation under sparse data. *Americas Conference on Information Systems*, 2004.

[16] Georg Lausen Ziegler, Cai-Nicolas. Analyzing correlation between trust and user similarity in online communities. *Second International Conference on Trust Management*, 2004.