A HYBRID RECOMMENDER: USER PROFILING FROM TAGS/KEYWORDS AND RATINGS

by

SWAPNIL NAGAR

B.E., Rajiv Gandhi Technical University, India, 2008

A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2012

Approved by:

Major Professor
Doina Caragea

Copyright

Abstract

Over the last decade, the Internet has become an involving medium and user-generated content is continuously growing. Recommender systems that exploit user feedback are widely used in e-commerce and quite necessary for business enhancement. To make use of such user feedback, we propose a new content/collaborative hybrid approach, which is built on top of the recently released hetrec2011-movielens-2k dataset and is an extension of a previously proposed approach, called Weighted Tag Recommender (WTR). The WTR approach makes use of tag information available in hetrec2011-movielens-2k, but it does not use explicit ratings. As opposed to WTR, our modified approach can make use of ratings to capture collaborative filtering and either user-tags, available in the hetrec2011-movielens-2k, or movie keywords retrieved from IMDB, to capture movie content information. We call the two versions of our approach Weighted Tag Rating Recommender (WTRR) and Weighted Keyword Rating Recommender (WKRR), respectively. Movie keywords (which are not user specific) allow us to use all ratings available in hetrec2011-movielens-2k, as WKKR associates the content information from movies with the users, based on their ratings. On the other hand, tags provide more specific information for a user, but limit the usage of the data to the user-movie pairs that have tags (significantly smaller number compared with all pairs that have ratings). Both our keyword and tag representations of users can help alleviate the noise and semantic ambiguity problems inherent in information contributed by users of social networks. Experiments using the WTRR approach on a subset of the dataset (which contains both ratings and tags) show that it slightly outperforms the WKRR approach. However, WKRR can be applied to the whole hetrec2011-movielens-2k dataset and results show that the information from keywords can help build a movie recommender system

competitive with other neighborhood based approaches and even with more sophisticated state-of-the-art approaches.

Table of Contents

List of Figures

List of Tables

Acknowledgments

Foremost, I would like to express my deep and sincere gratitude to my supervisor, Assistant Professor Dr. Doina Caragea of Department of Computing and Information Sciences, Kansas State University. Her understanding, wide knowledge, logical way of thinking, insightfulness and personal guidance have provided a good basis for the thesis. Her personal guidance, kind support and encouraging attitude helped constantly while working on this research and writing of my thesis. I could not have imagined having a better supervisor for my thesis.

Many thanks go in particular to my friend and project partner, Ana Stanescu, for her constant support and cooperation. I would also like to give special thanks to Dr. Susan Brown for supporting me as a GRA at the Bioinformatics Center.

I gratefully acknowledge my mother, Nisha Nagar, and father, Lalit Kumar Nagar, for their endless love, affection and concern.

At the end of my thesis, I also owe loving thank to my brother and all my friends who trusted me and made this thesis possible and an unforgettable experience for me.

Chapter 1

Introduction

As Web 2.0 applications continue to proliferate, the overabundant unstructured data that becomes available on the Internet contains great amounts of useful knowledge which consequently entails for automated information sifting. Overwhelmed by the huge number of options presented with, people rely more and more on the experiences of others for choosing movies, books and other products.

Recommender systems emerged in the mid-90s in order to filter out irrelevant information and select content that meets user needs. Burke [2002] has described recommendation systems as "An information filtering technology, that produces individualized recommendations as output or have the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options". These systems can be used for different purposes in several domains from offering products to consumer in e-commerce to finding proper information in research [Surowiecki and Silverman, 2007]. Surowiecki's *wisdom of crowds* (WOC) hypothesis states that when it comes to popular culture, among other domains, averaging the opinions of a large group of people captures the reality more accurately than legitimate experts. Such opinions are expressed online, and they constitute a great source of raw data that Recommender Systems (RS) [Resnick and Varian, 1997] can process in order to make suggestions for those who are seeking them.

An e-commerce recommender system will gather current customers information and past purchases and correlate these customers to products. With the use of some standard rec-

ommender algorithms, an e-commerce system provides customers with accurate recommendations. There are recommendation systems in different domains such as films, television programs, video, music, books, news, images, web pages [Adomavicius and Tuzhilin, 2005]. Many advanced systems try to help users to find right data according to their interest. Personalized recommendations are a key method for information retrieval. System like Amazon[1] help customers to find the best products online. Pandora[2] offers a wealth of information and services related to many aspects of music. It also helps users to find songs that they might be interested in by capturing the user behavior. MovieLens[3] tries to guide users to identify the movies they might like. YouTube[4] is known as the biggest collection of online videos and a system that can leverage the user browsing history for recommendation.

E-commerce websites utilize one or more recommender systems to suggests the best solution to their customers. Various movie businesses like Netfilx, IMDB, Hulu etc. recommend the movies by creating a relationship with the customer. Customer retention is very important to such websites, this relationship will often be to the benefit of the customer as well as the websites [Schafer et al., 1999]. Although there are several factors that influence the quality of recommender system [Kostakos, 2009], recommendations based on common viewpoints become more and more trustworthy and widely-used [Schafer et al., 1999].

The recommendation task is often times reduced to the problem of estimating what rating a user would give for an unseen item, or to finding a list of items that the user is most likely to enjoy. Movie recommendation is an open research area with unanswered problems and with growing social networking data. There is need of solutions to those problems to keep the customer en-tact.

Generally, recommendation systems can be categorized as content-based, collaborative or hybrid [Balabanović and Shoham, 1997], as described below:

[1]http://ww.amazon.com
[2]http://www.pandora.com
[3]http://www.movielens.org/
[4]http://www.youtube.com

- **Content-Based Recommender** (CBR) systems suggest items similar to the ones that the user has preferred in the past. However, there are some limitations to the CB technique, like the data scarcity problem. Modeling the user's interest is limited to extracting features from their browsing or purchasing history [Balabanović and Shoham, 1997]. Another limitation is that CB systems cannot identify new and different items that the user may enjoy, as it is prone to finding only those that are highly similar to the items in the history of that user.

- **Collaborative Filtering** (CF) systems [Su and Khoshgoftaar, 2009] filter large data sets in search for patterns and information of interest, by collecting preferences from multiple users. Collaborative recommendations are based on the user-user similarity. The system will recommend those items that are liked by users with matching taste. Similarly to the CB approach, CF techniques have shortcomings as well: a new item cannot be recommended until someone references it. Moreover, a user with unusual preferences may not receive recommendations unless there are other users that exhibit the same interests.

- A combination of the CBR and CF techniques, referred to as the **hybrid** approach, can alleviate some of the problems that each approach encounters individually, and can achieve superior results. Hybrid recommender system could offer good performance even with little or no user data.

Given this background, in this work we address the problem with collaborative filtering and content-based recommender. This work is an extension of prior work on hybrid systems by Liang et al. [2010]. Specifically, an extension of the Weighted Tag Recommender (WTR) and is meant to address the problem of tag synonym and semantic ambiguity. Their hybrid approach is a combination of collaborative and content based techniques and creates users and items profiles based on tags that are related with respect to an item. They manage to overcome the problems of personal tagging by extending the pool of user tags to related tags

that represent similar topics. Although the approach proved to be effective in improving the predictions, it has some limitations. Firstly, the algorithm does not consider explicit ratings while building a user or item profile, tags provided by the users are strictly considered as features of interest. In domains (e.g. movies, books) where you have both tags and ratings, it might be desirable to make use of the ratings of the users. It has been proven in the past that systems that leverage either the explicit ratings provided by the users, or implicit ratings inferred by the system have performed well (e.g. Netflix[5] that makes predictions based on user ratings and user habits). Secondly, no general item description is taken into consideration. In the work by Liang et al. [2010] they only consider item description specific to users. In case of movies, a more general description of the items can give more relevant features. Thus, features from item description can be significant when analyzed with ratings provided by users and can capture user behavior much accurately compared to user specific tags.

To alleviate the limitations faced by the Weighted Tag Recommender System that uses implicit ratings and only user specific tags, in this work we design an improved hybrid algorithm that make use of keywords extracted from content of movie or user tags, for user profiling and it also uses user ratings. Our recommender system constructs the topic preference for each user using keywords or tags. Similar to [Liang et al., 2010], our approach expands the tag/keyword pool by incorporating related keywords to alleviate the problem of synonymy and semantic ambiguity. We used a neighborhood approach to make recommendation based on both content user profile and a collaborative user profile. Movie recommender's are highly dependent on the prediction scheme. To find the best prediction scheme for our work, we study various prediction schemes.

[5]http://www.netflix.com

1.1 Thesis Contributions

To alleviate the limitations of the Weighted Tag Recommender approach [Liang et al., 2010], which does not incorporate user ratings, we consider a Weighted Tag/Keyword Recommender strategy that makes use of ratings, and thus exhibits a better way of generating user profiles. We explore the applicability of the neighborhood approach to classify users based on the properties of their closest neighbors in the feature space and use a hybrid filtering technique to generate recommendations. The users are modeled in terms of their preferences. The recommender system constructs the topic preference for each user based on tag/keyword weights that together with ratings can denote the degree of importance of those tags/keywords to that particular user. Then, by matching together users with similar opinions based on their profiles, each member of the system becomes part of a "neighborhood" of other like-minded users. Movie recommenders are highly dependent on their prediction scheme. Therefore, in this work, we also study various prediction schemes in order to find an optimal one for making recommendations in the movie context.

In summary, the main contribution of this thesis is improving the results of a hybrid algorithm for movie recommendation. Injecting features from a consolidated movie content database (such as tags or IMDB keywords) and profiling based on ratings, is shown to enhance recommendations. Our approach uses a technique similar to the one proposed in [Liang et al., 2010] which expands the keyword/tag pool by incorporating related keywords/tags to alleviates the problem of synonymy and semantic ambiguity.

Chapter 2

Background on Recommender Systems

E-commerce websites help consumers to discover products of interest from huge data available over the Internet. These websites use recommender systems to generate personalized suggestions to consumers on how to find relevant items from the large number of choices. In the past decade, lot of work has been done on recommender systems and various techniques have been developed. This chapter aims to discuss general concepts and terminology on recommendation techniques. We discuss, in more detail, major types of recommendation techniques and algorithms, pointing out their advantages and disadvantages.

2.1 Recommender Systems

It is estimated that by the end of year 2012, one third of the world population will be using the Internet and about 1.2 billion people will join social networking websites. People spend 6.7 billion hours on social networking websites in a month, constantly pouring the information on the web[1]. As a result, we are drowning in information, but continue to starve for knowledge [Miller, 2007]. Thus, it is becoming difficult for people to pick the resources of their interest. Many techniques and systems have been developed in the last decade to help users to find the right information. One of the most successful technologies is known as

[1] http://www.internetworldstats.com/stats.htm

Recommender Systems, which are based on personalized information filtering, and predict relevant items for a particular user.

Recommender systems leverage the community opinions to identify content of interest to an individual from an overwhelming set of choices [Resnick and Varian, 1997]. Recommender systems generate item sets, that aim to be highly customized to specific users. Therefore, it can be said that recommender systems perform a mapping between items of interest and users [Mobasher et al., 2007].

Given the enormous amount of data and diverse users, recommender systems can not simply rely on users or community for recommendations. Ideas from different areas such as Natural Language Processing, Artificial Intelligence, Human-Computer Interaction, and Information Retrieval are incorporated in advanced systems for better predictions. During the past decade significant progress has been made in the field of recommender systems, though this is still a very active and popular area, as there are many open ended problems that need to be addressed. To choose the right recommendation algorithm for a problem, offline experiments need to be conducted over the same set of real data. Typically, recommender systems are broadly classified in three categories:

1. **Content-based recommendations:** Items recommended to the users, similar to the ones the user preferred in the past;

2. **Collaborative recommendations:** Items recommended to the users, by opinion from the people with similar tastes and preferences;

3. **Hybrid approaches:** Hybrid approaches combine one or more collaborative and content-based techniques.

   We will describe these methods in more details in the next sections.

2.2 Content Based Methods

Content-Based recommender (CBR) systems suggest items from huge available options based on similarity between item features and user's preference [Van Meteren and Van Someren, 2000]. Typically, these recommender systems suggest items similar to the items preferred in the past by the user. In content-based systems, the items are represented by their associated features. The algorithm compares the collective user information against the feature content of a new item, and items with high similarity score are recommended. CBR approaches construct a user profile from the features used to describe the items that the user has rated [Burke, 2002]. Decision trees, neural nets, and vector-based representations can be used to construct the proof depending on the problem domain and dataset. CBR models are ever evolving models which keep updating the user profile based on user item preference and activity.

Sometimes, CBR approaches rely on users feedback to learn their (users) preference. Typically, information about user choices or interaction with the recommendation system, can be used to construct the user profile. User activity and user queries may help recommender systems to filter out the items that are already viewed by the user. The information needed to interpret user feedback can be learned in two ways, through implicit feedback or through explicit feedback [Gauch et al., 2007].

Explicit feedback: On various websites, users are asked to provide general information about themselves this type of information collected from users directly is considered to be explicit ratings. Such information includes: age, gender, location, interests, etc. Another option that most websites opt for is to ask feedback on a product. Users are suppose to provide feedback either in the form of surveys or general forms. This technique is bothering and most of the time users are not interested in filling out the forms, thus the resulting profiles are imprecise. Hence, nowadays e-commerce websites are requesting users to express their opinions by selecting a value in a range of explicit ratings. This is usually less troublesome to users than filling out the forms.

Implicit feedback: Monitoring the user activity over the web can provide implicit feedback. Usually, users are not aware that they are providing feedback to the system. One such example is YouTube, where direct relation between the amount of time spent on a single video per session and user interest can be captured. This type of feedback may not be as accurate as the feedback in the form of explicit ratings by users, but users are not disturbed.

As discussed earlier, CBR systems work best when recommending mostly text-based items, where the content is extracted in form of keywords. One such popular content-based technique is the Fab system [Balabanović and Shoham, 1997], which is designed to recommend the web pages to the users. In the Fab system about 100 most important keywords are chosen to represent the web pages in the corpus. Similarly, documents are represented by 128 most distinct and frequent words in [Pazzani and Billsus, 1997] and the importance of each word in a document is measured by some weighting scheme.

There are many weighting schemes to calculate the importance of a keyword, but the term frequency/inverse document frequency (TF-IDF) measure is one of the most popular and known measures for specifying keyword weights. TF correlates to the term's frequency, defined as the number of times term t appears in the currently scored document d ($TF_{t,d}$). Documents that have more occurrences of a given term receive a higher score. Normalized term frequency is introduced to capture more accurately the importance of a term t in document d. Normalized TF is a ratio between the frequency of a terms occurring in a document and the maximum term frequency in that document ($tf\text{-}max(d)$). Thus, $nTF_{t,d}$ is normalized term frequency of a term t in a document d and is given by:

$$nTF_{t,d} = \frac{TF_{t,d}}{tf - max(d)} \tag{2.1}$$

A high nTF weight does not always imply that the term or keyword is discriminative for the document. However, IDF ensures that a high score is assigned to a term or keyword which is discriminative for the document and a low score is assigned to a term or keyword

which is highly used in all the documents [Papineni, 2001]. Therefore, IDF is often used in combination with nTF. The inverse document frequency for a term t is given as:

$$IDF_t = log\frac{N}{n_t} \tag{2.2}$$

where N is the total number of documents in the corpus and n_t is the number of documents in which term t appears. TF-IDF weight for term t in document d is defined as:

$$w_{t,d} = nTF_{t,d} \times IDF_t \tag{2.3}$$

To capture the TF-IDF in the content based component of our approach, we have used the modified version of TF-IDF which moves one step further and eliminates the noise and synonymy problems.

CBR techniques have been succesfully used in many areas, including in the field of biological and medical sciences. PURE [Yoneya and Mamitsuka, 2007] is an article recommendation system, where a user has to feed initial preferred articles into the system, which then iteratively captures the user preferences from their inputs. In machine learning and information retrieval, VSM is generally used to create item and user profiles [Huang, 2008]. Rocchio [1971] proposes a VSM based technique, where performance of the system improves over the time, as the system collects more and more knowledge from user feedback.

Advantages of content-based recommender systems:

- Implicit feedback from users is enough to construct the user profile and with increase of database content over the time, performance of CBR gradually improves.
- CBR can help to recommend new or unpopular items to users based on their taste. Hence, new items do not starve for users explicit feedback.

Disadvantages of content-based recommender systems:

- CBR systems are limited to features that are explicitly associated with items. For better performance of the system, a sufficient set of features is required, so either

automatic feature extraction from content is needed or manually annotation of items is required. Applications of feature extraction methods are somewhat limited to text based approaches and harder to apply in domains such as image or video recommendation. Also, it is impractical to identify features manually due to limitation of resources [Shardanand and Maes, 1995].

- Another problem with CBR is that when items are represented by the same set of features, they can be indistinguishable.
- CBR systems suffer from an insufficient number of ratings at very early steps.
- Over-specialization can occur when the CBR system only recommends items scoring highly against a user's profile. In such cases, the user is restricted to seeing items similar to those already rated. Often this is addressed by injecting some randomness in the predictions.
- New users may not be able to get accurate predictions according to their taste. For better predictions, users need to have a sufficient number of ratings, which is not always possible.

2.3 Collaborative Filtering

Collaborative filtering (CF) is a technique for producing personalized recommendations by computing the similarity between the current user and other users with similar choices. Thus, the current user choice is predicted by gathering choice information from other users with similar preferences. If choices matched in the past, it is assumed that they will match in future as well. For accomplishing this task, a large amount of data is processed [Schafer et al., 2007].

Prediction and recommendation are the two main parts of a CF [Walia, 2008]. Collaborative prediction uses the current preferences that are available and the other users preferences

relation to predict the current user preferences. Developing a set of items which are more likely to be of interest to the current user is known as collaborative recommendation. For example, CF uses item preference database to suggest items to new users. Typically, there is a set of n users given by $U = \{u_1, u_2, ..., u_n\}$ and set of m items given by $I = \{i_1, i_2, ..., i_n\}$, and each user $u_j \in U$ has a set of items I_{u_j} that the user u_j has rated. Usually, ratings are in the range 1 to 5. CF leverages this user rating data and implicitly learns the user preference [Miller et al., 2004]. A user-item ratings matrix is a list of users and the items they like or dislike, specified using ratings. The CF task is to predict values in this matrix, specially, predict values for which the user did not provide ratings and, then recommend the items with high prediction scores.

2.3.1 Types of Collaborative Filtering Algorithms

Collaborative filtering algorithms can be divided into two groups [Shani et al., 2002]:

1. **Memory-based collaborative filtering algorithms:** Memory-based algorithms exploit the entire item-user database. A set of similar users are identified for the current user, and rating predictions are generated based on ratings in the neighborhood of the current user. Memory-based CF is easy to implement and new data addition is easy to add incrementally. In the memory-based category, the most popular non-probabilistic approach is the k Nearest Neighbor algorithm (kNN). kNN can identify user-based nearest neighborhoods and item-based nearest neighborhoods, where either users or items are represented using ratings. The disadvantage of memory-based CF techniques is that they solely depend on user ratings, and a decrease in performance is observed when data is sparse, and also for new users/items, the prediction is difficult. Such techniques exhibits poor scalability for larger dataset. For sparse dataset, dimensionality-reduction methods like SVD are recommended as they generate better predictions.

2. **Model-based collaborative filtering algorithms:** Model-based CF algorithms

use the pure rating dataset to construct a model to make prediction [Breese et al., 1998]. Well known model-based techniques include Bayesian models which use the naive Bayes strategy to make predictions. Clustering CF chunk the big dataset in a set of clusters, then recommendations are made independently for each cluster to achieve better scalability [Ungar and Foster, 1998]. Model-based CF methods can deal with the sparsity, scalability and other problems in a better way than memory-based methods. The prediction performance is improved and an intuitive rationale for recommendations is given. Irrespective of several advantages, this technique also has some shortcomings. Model building is usually expensive, furthermore model-based CF performs a trade-off between scalability and performance prediction. At last, useful information can be lost when dimensionality reduction techniques are employed.

2.3.2 Similarity Measures

Similarity computation is the backbone of collaborative filtering as neighborhood formation is done based on these values. To evaluate the similarity between any two users or items, cosine-based similarity is modified. In general, cosine similarity can estimate the similarity between two documents, where each document is a vector of words and their weights (e.g. TF-IDF weight). In recommender systems, instead of documents, we have users or items, and instead of word frequencies we have ratings. Thus, cosine similarity is defined as:

$$sim(u,v) = \frac{\sum_{i \in I} r_{u,i} \cdot r_{v,i}}{\sqrt{\sum_{i \in I} r_{u,i}^2} \cdot \sqrt{\sum_{i \in I} r_{v,i}^2}} \tag{2.4}$$

where I is set of items that both users u and v have rated and $r_{u,i}$ is the rating given by user u to the item i, otherwise it is zero. Computing similarity using basic cosine measure in user-based case has one important drawback:- the differences in the rating scale between different users are not taken into account [Sarwar et al., 2001]. To overcome their drawback in cosine similarity, many researchers rely on Pearson Correlation Coefficient (PCC) and its

variants to calculate the item-item and user-user similarity. PCC measures the degree to which two vectors are linearly related with each other. User-user PCC is denoted as $w_{u,v}$ and is given by:

$$w_{u,v} = \frac{\sum_{i \in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i \in I}(r_{v,i} - \bar{r}_v)^2}} \tag{2.5}$$

where I is the set of items that both users u and v have rated; $\bar{r}_u$ is the average rating to items by user u. To calculate item-item PCC for items i and j, first, the set of common users (users who rated both items) $u \in U$ is found. Item-item PCC is denoted by $w_{i,j}$ and given by:

$$w_{i,j} = \frac{\sum_{u \in U}(r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U}(r_{u,i} - \bar{r}_i)^2}\sqrt{\sum_{u \in U}(r_{u,j} - \bar{r}_j)^2}} \tag{2.6}$$

where $r_{u,i}$ is the rating of user u on item i, and $\bar{r}_i$ is average rating of item i by user u.

2.3.3 Prediction Computation

Rating prediction is the most important step in a collaborative filtering system. In the nearest neighborhood techniques, the set of nearest (similar taste) neighbors of an active user is determined based on similarity scores and weighted aggregate of neighbor ratings used to predict the rating of the users.

Simple Weighted Average: This is a basic rating prediction scheme, which uses simple weighted average to predict the rating [Sarwar et al., 2001]. $P_{u,i}$ denotes the predicted rating for user u on item i and is given by:

$$P_{u,i} = \frac{\sum_{j \in N} r_{u,j} w_{i,j}}{\sum_{j \in N} |w_{i,j}|} \tag{2.7}$$

where N is the set of all items that the user has rated, other than item i, $w_{i,j}$ is the similarity score between item i and item j, and $r_{u,j}$ is the rating for item j by user u.

Weighted Sum of Users Ratings: This prediction scheme is a modified version of (2.7), which uses the mean weighted average of all the ratings on the active item. It is defined as:

$$P_{u,i} = r_u + \frac{\sum_{v \in U} (r_{v,j} - \bar{r}_v) \cdot w_{u,v}}{\sum_{v \in U} |w_{u,v}|} \tag{2.8}$$

where u is the active user and the rating calculation is for item i; $v \in U$ represents users that have rated item i, $w_{u,v}$ is the similarity score between user u and user v; $r_{v,i}$ is the rating for item i by user v; $\bar{r}_u$ and $\bar{r}_v$ are the average rating for user u and user v, respectively.

Apart for these, many researchers have modified the prediction schemes or develop variants of the existing ones, but they all have the same purpose to rate predictions. In our work, we use user-mean-centered rating scheme (2.9) which is the most reliable and effective prediction scheme in the movie domain:

$$\hat{r}_{(u,m)} = \bar{r}_u + \sigma_u \frac{\sum_{v \in U} \frac{w_{uv}(r_{v,m} - \bar{r}_v)}{\sigma_v}}{\sum_{v \in U} |w_{uv}|} \tag{2.9}$$

In Equation (2.9), $\bar{r}_u$ is the average of the ratings given by user u, w_{uv} is the similarity value between user u and user v, σ_u is the standard deviation of ratings given by user u and all the other users in the corpus.

Advantages of Collaborative Filtering Recommenders:

- Collaborative filtering can perform well in cases where there is not much data or content associated with items. It also performs well in cases where the data associated with items is hard to analyze or have inconsistencies, e.g. in ideas, opinions, etc.

- Recommender systems built solely on collaborative filtering approach have the ability to provide recommendations that are relevant to the user, but do not contain content

from the user's profile.

Disadvantages of Collaborative Filtering Recommenders:

- New User: There needs to be enough other users already in the system to find a match for a particular user.
- Sparsity: Most users do not rate most items and hence the user-item matrix is typically very sparse. If there are many items to be recommended and even if there are many users, because the user-ratings matrix is sparse, it is hard to find users that have rated the same items.
- First Rater: It is not possible to recommend an item that has not been rated before. This problem comes for new items mostly. An obscure item may also face this problem.
- Popularity Bias: Collaborative filtering cannot recommend items to someone with unique taste. In general, there is a tendency to recommend the popular items.

2.4 Hybrid Techniques

In general, hybrid recommenders are systems that combine multiple recommendation techniques together to achieve better performance and to eliminate disadvantages in recommendation system. Hybrid systems aim to take advantage of all techniques (that are combined together) and obtain more accurate predictions. Most popularly, researchers combine collaborative filtering with some technique to avoid new-user problem. Hybrid techniques perform well in some domains but cannot be generalize for all recommendation problems. Hybrid techniques mainly depend on the problem domain and data, as different types of combinations produce different results. Some of the hybridization techniques are mentioned below:

2.4.1 Weighted Recommenders

A weighted hybrid recommender score for an item is computed of the output from all available recommendation techniques. For example Claypool et al. [1999] combine the outputs or ratings obtained from content and collaborative filtering systems into one final (hybrid) recommendation using linear combination of ratings. Equal weightage is given to both techniques. Another popular recommender system in this category is [Pazzani, 1999] hybrid combinator, which evaluates the output from each recommender as a set of votes (defined, undefined and neutral), and then combines the sets to get general consensus.

2.4.2 Switching Recommenders

Switching hybrid systems incorporate the item level sensitivity into the hybridization strategy. Based on some criteria the system switches between the recommendation techniques. For example if the content-based system cannot generate a relevant result, then collaborative filtering is attempted. However, finding the right criteria for switching adds additional complexity to the process. Given that content and collaborative recommender systems can't deal with the new-user problem, it is obvious that switching systems also fail to address the new-user problem. The benefit of such a system is that it can be sensitive to the strengths and weaknesses of its constituent recommenders.

2.4.3 Mixed Recommenders

Mixed recommenders help to make a large number of recommendations in less time. This type of recommenders take the consensus from two or more sources and ranks the items using a complex combination technique. The advantage of using the mixed recommender system is that it alleviates the new user problem.

2.4.4 Cascaded Recommenders

Unlike other hybridization techniques, the cascade hybrid technique is a staged process. In this technique, the first recommender technique generates a ranked candidate set and then a second recommender refines the recommendations in the candidate set. The second step in cascading is to identify poorly rated items and to focus only on those items for which additional discrimination is needed. The cascade hybrid system works well, especially when combining two components of differing strengths. Thus, it is more efficient than weighted hybrid that employs all the techniques for all items.

2.4.5 Feature Augmentation Recommenders

In this technique, the first recommeder produces a rating or classification of the items and this information is fed to the next recommender. In general, features extracted from the first recommender is used by the next recommender. Augmentation is attractive because it offers a way to improve the performance of a core system.

Chapter 3

Related Work

This chapter discusses the related work relevant to the problem addressed in this thesis. We focus on related work in the movie recommendation domain and point out advantages and disadvantages of various methods. Specifically, we consider approaches using tags with ratings, and keywords with ratings, in what follows.

The work on recommender systems has been expanding greatly and is constantly improving. Such recommender systems are currently applied to a wide range of domains, from entertainment items to scholarly articles, from products to friend suggestions in social networks. The culminating point which attracted a lot of interest and also the attention from the media was the announcement of the million-dollar prize from Netflix [Bell et al., 2008], which required a 10% improvement over their best recommendation technique at that time. For more comprehensive information on recommender systems, the reader is referred to [Marinho L.B., 2011].

Tagging is a type of labeling, whose purpose is to assist users in the process of finding content on the web. It has evolved considerably thanks to social networks and has become a very popular concept. In 2004, Thomas Vander Wal assigned the name "folksonomy"[1] to the tag system developed by Web 2.0 consumers, as a derivation from the phrase "people's taxonomy". Although the tagging terms are highly personalized, their aggregation conveys a sound basis for prediction algorithms. For example, Said et al. [2012] propose a folksonomy-

[1] http://vanderwal.net/folksonomy.html

based approach to personalize tags. For each user, each tag is assigned a value obtained from averaging the ratings the user gave to the movies tagged with that particular tag.

Tagging data is used with existing CBR methods to improve the overall predictive accuracy of the algorithm. The work by De Gemmis et al. [2008] capture the user preference from both item description and tagging data. This approach profiles a user by considering tag information as an additional source. Experimental results confirm the improvement in prediction accuracy.

Recently, tagging data has also been exploited with traditional collaborative filtering to enhance the performance. In [Wang et al., 2010], tags are used to build profiles for users, then similarity scores are used to determine like-minded neighbors. This approach uses implicit ratings to profile items or users. Explicit ratings are used only for predicting missing ratings. The dataset used in this work is a subset of the MovieLens[2] dataset. Preprocessing was performed to remove noise and ambiguity from tags. But [Wang et al., 2010] fail to handle the problem of tag synonymy and tag ambiguity. In reality, it is quite impossible to manually remove such noise from data and tags considered as a noise in one dataset can have great significance in some other dataset. Thus, methods that can handle noise in the dataset by themselves are highly desirable.

Tags are free annotations and there are no constraints enforced, which makes tag-based recommendations suffer from degraded performance because of semantic problems, like polysemy and synonymy [De Gemmis et al., 2008]. A hybrid system proposed by Liang et al. [2010] that deals with these problems is based on weighted tags and was developed to recommend books from the Amazon database. Liang et al. [2010] manage to overcome the problem of personal tagging by extending the pool of user tags to related tags that represent similar topics. Although the approach proved to be effective in improving the predictions, it has some limitations. In the first place, while building the user and item profiles, the algorithm does not consider explicit ratings, but implicit ratings: if a tag has been assigned, then it

[2]http://www.grouplens.org/node/12

is inferred that the user is interested in the item. However, that may not always be the case. Users may still tag movies that they do not like, in which case, the rating holds the information about their true preference. For domains (e.g., movies, books, products) where both tags and ratings are available, a recommender system should exploit all the information available.

Systems that leverage ratings – which can be either explicitly provided by the users or implicitly inferred by the system – are known to perform well, for example Netflix [Bell et al., 2008]. By integrating a weighted combination of rating and tag information from social networks can improve recommendations [Clements et al., 2010]. However, ratings are not always noise free [Amatriain et al., 2009], therefore, we combine them with more solid features (i.e., keywords) extracted from movie descriptions.

The following approaches make use of the hetrec2011-movielens-2k dataset, which we also use in this paper:

The system proposed by Bothos et al. [2011] is an ensemble of various recommenders, called Information Market Based recommender Fusion (IMBrf), primarily used for mining and aggregating the information from various sources. This technique is inspired from the market, where information from heterogeneous sources is incorporated to make predictions about future events. We compare our results with the results of other approaches as reported in [Bothos et al., 2011], including a pure collaborative filtering technique (CF) and a content-based recommender system, called content analysis (CA). For CF, the authors used the neighborhood based approach and set the size of the neighborhood to 30. We will preserve these settings in our experimental setup, to be able to make fair comparisons. The CA recommender is based on latent topic analysis, and movies are mapped to topics via tags. The prediction is made by finding topics in new movies that are correlated to the user profiles. Another recommender, based on averaging the ratings (AVGR), is presented in [Bothos et al., 2011], where an unrated item's rating is estimated from the weighted average of other ratings from other users. Finally, Linear Least Square (LLS) is proposed in [Bothos et al.,

2011]. LLS is a linear combination of CF, CA and AVGR: $R_{LS} = \alpha CF + \beta CA + \gamma AVGR$, where R_{LS} denotes the predicted rating (the parameters α, β and γ are found by optimization, for more details see [Bothos et al., 2011]). In [Jones et al., 2011], the authors propose learning multiple models which can incorporate different types of inputs to predict the preferences of diverse users. Probabilistic Matrix Factorization (PMF) is a variational Bayesian inference technique, used to alleviate the overfitting problem in singular value decomposition (SVD) approaches. Priors are introduced and parameters are estimated using variational Bayesian inference [Nakajima and Sugiyama, 2007]. PMF models the user preference matrix as a product between the lower-rank user and movie matrices [Jones et al., 2011].

k-NN is one of the most popular methods, and also a simple and easy to implement approach. k-NN is computationally efficient and stable with the addition of new users or items. Another important strength is its serendipity that gives users unexpected prediction, yet interesting. k-NN cannot overcome the data sparsity problem. However, with the right combination of models, this problem can be overcome. Also, various approaches strongly advice not only to use content-based information to improve the active user knowledge, but also leveraging the explicit feedback from users at collaborative level. This can be achieved by hybrid models where all the components are represented under the same formalism. Usually, at the feature level, implicit ratings are used to profile users or items which can lead to the over-specialization problem. Data sparsity along with relations between features is still an open problem in recommender systems. However, researchers still struggle to discover a perfect way to incorporate the ratings with the tags or keywords.

In this thesis, we proposed a method that makes use of both tags and keywords along with ratings. We study the behavior of the algorithm when applied to a movie dataset. Experiments show that, recommendations made using the neighborhood approach are accurate and comparable to state of the art results.

Chapter 4

Problem Discussion and Approches

In this chapter, we first discuss the problems addressed by [Liang et al., 2010] in Section 4.1 and then we provide an overview of the Weighted Tag Recommender approach and of the proposed approach Weighted Keyword Recommender (WKRR) and Weighted Tag Rating Recommender (WTRR) algorithms and explain how we implemented these algorithms in Section 4.2. Further in Section 4.3 the neighborhood formation and the hybrid recommendation generation are discussed. Finally, we define efficient prediction scheme in Section 4.4.

4.1 Problem Addressed

The book recommender system proposed by Liang et al. [2010] is built from tag information only. The authors state that tags can capture the content information of items. However, tags are sometimes meaningful only to the users that assigned them. They can be ambiguous and can also have a lot of synonyms. Liang et al. [2010] developed a way of addressing these problems by expanding the tag set that is relevant to a user to include other related tags. They construct user profiles from these weighted tags. Then, based on the same tag expansion procedure, they build item preference profiles, which include other related tags that are relevant for describing an item. In this approach, item recommendations are made by combining a user based collaborative approach with a content based approach. However, the user ratings are never explicitly used. Instead, implicit ratings are used. The implicit

ratings are obtained by assuming that a user who tags a movie likes that movie.

We expand the idea of weighted tags in the context of movie recommendations. Given that the hetrec2011-movielens-2k data shows that tags may not always capture the true preferences of users, in our scenario, we incorporate the actual ratings. One main difference in our approach is that, instead of simply counting the number of times a user u_i has tagged an item with the tag t_x (as done in [Liang et al., 2010]), we add up the ratings that user u_i has assigned to the movies tagged with t_x. Furthermore, we have observed that the number of tagged movies is significantly smaller than the number of rated movies. To be able to capture all rating information, we propose to use collective user keywords associated with a movie, available at IMDB. IMDB allows users to provide keywords in a controlled manner, thus keyword descriptions of movies can be considered as consolidated "word of mouth". Our intuition is that such information can produce better recommendations than ratings alone, when tags are not available, as it can capture content features. Thus, our approach works either with user-specific tags or collective user keywords, and actual ratings, as opposed to the original approach that considers only user-specific tags and implicit ratings.

4.2 Approaches: WTRR and WKRR

Before describing our approaches, we introduce some notations. The user set $U = \{u_1, u_2, ..., u_{|U|}\}$ contains all the users that tagged movies in hetrec2011-movielens-2k dataset. The movie set $M = \{m_1, m_2, ..., m_{|M|}\}$ contains all movies from the corpus, the tag set $T = \{t_1, t_2, ..., t_{|T|}\}$ contains all the tags used by the users in U to label movies in M. The keywords set $K = \{k_1, k_2, ..., k_{|K|}\}$ contains all the keywords used to annotate movies. Finally, we denote by $R = \{0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5\}$ the set of all possible ratings that users can give.

Our approach falls in the neighborhood category. To make recommendations, we represent user profiles using two components. First, we consider a movie tag/keyword based component, denoted by u^T and u^K, respectively. The u^T profile is a vector with T elements, while the u^K profile is a vector with K elements. Both of them are meant to capture

movie content, in other words topic preference. Second, we consider a movie rating based component denoted by u^M. The u^M profile is a vector with M elements and captures the collaborative filtering idea.

With these notations and preliminaries, we will present the details of our WTRR and WKRR approaches in what follows. We start by showing how to construct the content and collaborative profiles for a current user, followed by details about how to find the user's neighborhood using the profiles, and, finally, how to make predictions for the current user based on the information in the neighborhood. To illustrate the concepts defined in our approach we will use the user-movie-tag-rating graph in Figure 4.1 and the user-movie-keyword-rating graph in Figure 4.2

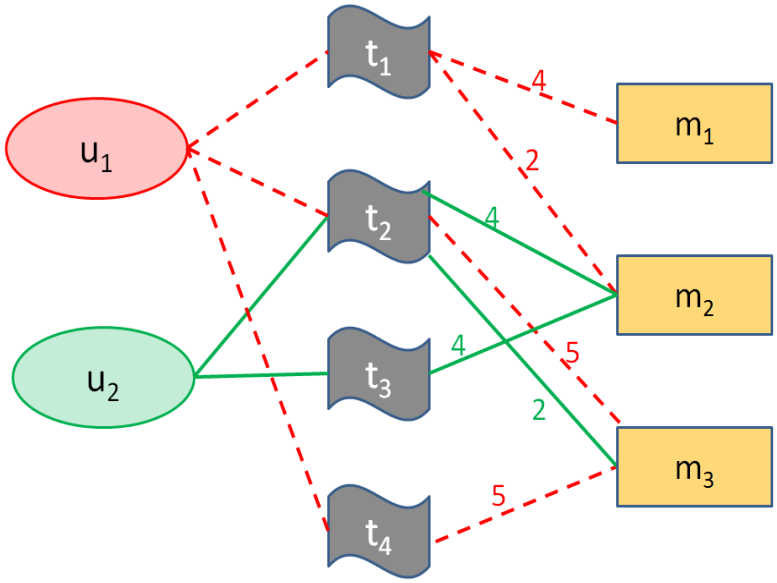

Figure 4.1: *Example of user-movie-tag-rating graph, used in WTR and WTRR approaches*

Figure 4.1 shows a scenario where we have 2 users and 3 movies. In this example, user 1 has rated and tagged movies m_1, m_2 and m_3 with ratings 4, 2 and 5, respectively. User u_2 has rated movies m_2 and m_4 with ratings 4 and 2. Edge from tag to movie is associated with a rating which is inherited from the users to avoid any confusion.

Figure 4.2, shows the keyword based scenario, where our task is to construct the user profile based on keywords and ratings. In this figure, movie m_1 has keywords k_1 and k_4,

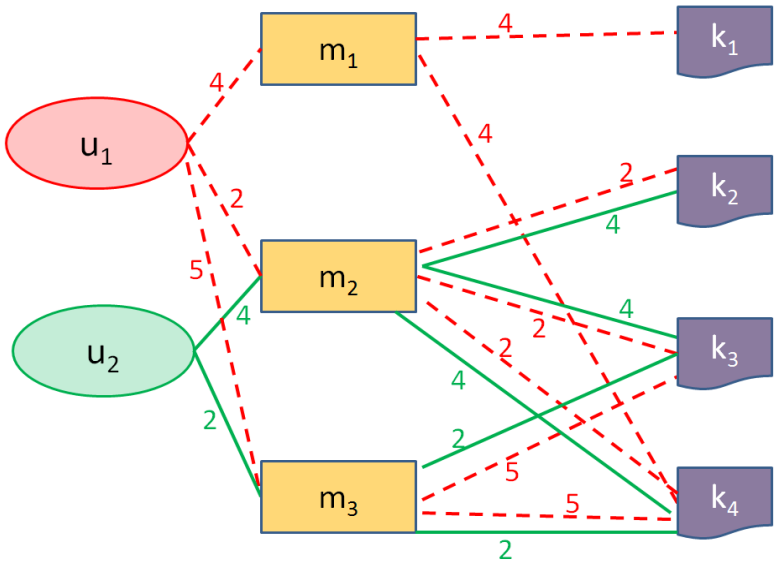

Figure 4.2: *Example of user-movie-keyword-rating graph, used in WKRR approach*

movie m_2 has keywords k_2, k_3 and k_4 and lastly movie m_3 contains keywords k_3 and k_4. For better visualization of actual data, Figure 4.3 is a subgraph specific to user u_2.

4.2.1 User Profiles

There are three steps that we need to consider in order to construct a tag/keyword user profile. First, we calculate the relevance of a tag or keyword to a movie as a weight. Using such weights, we next estimate relatedness between two tags/keywords, and finally use the relatedness information to construct user profiles.

Relevance of a Tag/Keyword to a Movie

WTR: Before describing how tag/keyword relevance weights are calculated in our approach, we first show how the *movie tag relevance weight* is calculated in the original WTR approach [Liang et al., 2010]. Let m_i be a movie from M and let T_{m_i} be the set of all tags used by different users to describe the movie m_i. For each tag t_x from T_{m_i}, the *movie tag relevance*

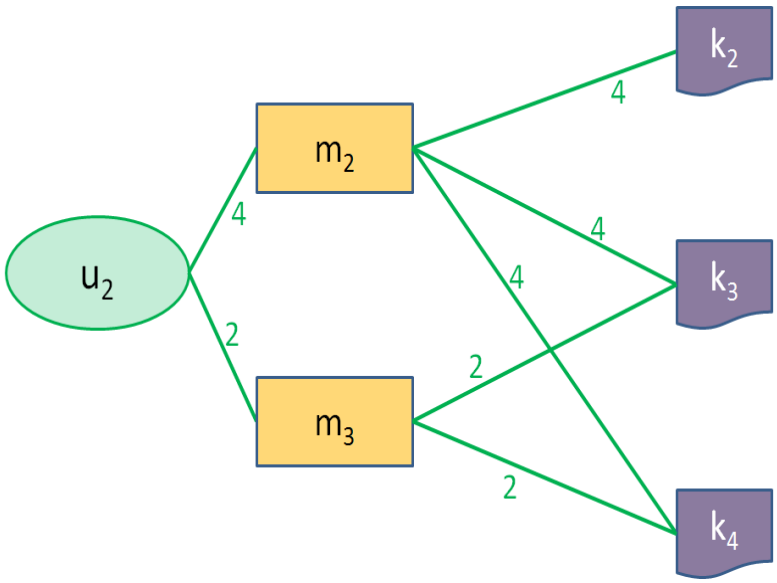

Figure 4.3: *Simplified keyword graph of user u_2 for WKRR approach*

weight is defined as $w_{m_i}(t_x)$ and is calculated using the equation:

$$\text{WTR based: } w_{m_i}(t_x) = \frac{n_{m_i,t_x}}{\sum_{t_y \in T_{m_i}} n_{m_i,t_y}} \tag{4.1}$$

where n_{m_i,t_x} represents the number of times the tag t_x has been used by the users in the corpus to describe the movie m_i. The value of $w_{m_i}(t_x)$ signifies how popular the tag t_x is for the movie m_i. This relevance metric reflects the *wisdom of crowds*. In other words, the higher the value of $w_{m_i}(t_x)$, the better the tag t_x represents the topic of the movie m_i.

With reference to Figure 4.1, we calculate the tag relevance metric for movie m_3. In general relevance weight is estimated for all the tags. For movie m_3, only tags t_2 and t_3 are used for tagging. Hence, we have $n_{m_3,t_2} = 2$ and $n_{m_3,t_4} = 1$ and $n_{m_3,t_1} = n_{m_3,t_3} = 0$. Hence, $w_{m_3}(t_1) = \frac{0}{1+2} = 0, w_{m_3}(t_2) = \frac{2}{1+2} = .667, w_{m_3}(t_3) = \frac{0}{1+2} = 0$, and $w_{m_3}(t_4) = \frac{1}{1+2} = .334$

WTRR: As our goal in the WTRR approach is to capture ratings, in addition to tags. The modified movie tag relevance weight Equation (4.1) is based on ratings rather than simple counts. To be able to use both ratings and tags, we must ensure that the user who tagged a movie, also rated that movie. In other words, a movie must be both tagged and rated by a particular user. The adapted formula for calculating movie tag relevance weight

using the WTRR approach is:

$$\text{WTRR based: } w_{m_i}(t_x) = \frac{\sum_{u_j \in U_{m_i,t_x}} r_{u_j,t_x}(m_i)}{\sum_{u_j \in U_{m_i}, t_y \in T_{m_i}} r_{u_j,t_y}(m_i)} \tag{4.2}$$

where the numerator is a summation of the ratings $r_{u_j,t_x}(m_i)$ assigned to the movie m_i by all the users u_j who used t_x to annotate it. The set of users who used t_x to tag m_i is denoted by U_{m_i,t_x}. The denominator represents a summation of all the ratings from the users who tagged m_i. The value of $w_{m_i}(t_x)$ now captures the true popularity of the tag t_x with respect to a movie m_i.

Based on the Equation (4.2) we calculate the relevance metric for movie m_3 using WTRR approach. For this instead of counting the occurrence of tags, we will incorporate the direct user ratings. So, $\sum_{u_j \in U_{m_3,t_2}} r_{u_j,t_2}(m_3) = 2 + 5 = 7$ and $\sum_{u_j \in U_{m_3,t_4}} r_{u_j,t_4}(m_3) = 5$ and other sums will be zero. Hence, $w_{m_3}(t_1) = \frac{0}{2+5+5} = 0, w_{m_3}(t_2) = \frac{2+5}{2+5+5} = 0.5833, w_{m_3}(t_3) = \frac{0}{2+5+5} = 0, w_{m_3}(t_4) = \frac{5}{2+5+5} = 0.4166$

WKRR: Similar to tags, we can also calculate the relevance of a keyword with respect to a movie as a weight. However, as opposed to tags that are specific to users, keywords are not associated with specific users. Here, we want to capture how relevant a keyword k_x with rating r is to a movie m_i. For example, how relevant is "gun" as a preferred keyword to the movie "Matrix", as opposed to how relevant "gun" is as a disliked keyword to the movie "Matrix". Thus, this measure will capture the community's amenity/approval of a keyword toward a movie. The movie keyword relevance metric $w_{m_i}(k_y, r)$ is given by:

$$\text{WKRR based: } w_{m_i}(k_x, r) = \frac{n_{m_i,k_x,r}}{\sum_{k_y \in K_{m_i}} n_{m_i,k_y,r}} \tag{4.3}$$

where m_i is a movie from M, K_{m_i} is the set of all the keywords used to represent movie m_i, $n_{m_i,k_y,r}$ is the count of how many users rated the movie m_i which contains keyword k_y with rating r. For example, if k_x belongs to a movie that the user u_i rated with 4.5, $n_{u_i,k_x,4.5}$ is the count of movies sharing keyword k_x which have been rated with 4.5.

Now, we demonstrate the calculation of $w_{m_3}(k_4, 4)$ for user u_2. First, we use Equation (4.3) to calculate the movie keyword relevance metric for movie m_2. Practically, we will be having 10 different matrices for a movie specific to each rating. We show the calculation for $w_{m_2}(k_4, 4)$. For this, we calculate $n_{m_2,k_4,4} = 1$, $n_{m_2,k_2,4} = 1$ and $n_{m_2,k_3,4} = 1$. Then, $w_{m_2}(k_4, 4) = \frac{n_{m_2,k_4,4}}{n_{m_2,k_4,4}+n_{m_2,k_2,4}+n_{m_2,k_3,4}} = \frac{1}{3}$. Similarly, $w_{m_3}(k_4, 2) = \frac{n_{m_3,k_3,2}}{n_{m_3,k_4,2}+n_{m_3,k_3,2}} = \frac{1}{2}$.

Tag Relatedness Metric for a User

Given the relevance of a tag or keyword with respect to a movie, we can calculate the relatedness of two tags or keywords with respect to a user, as explained below. The relatedness metric is used when constructing the user profiles, in order to avoid semantic ambiguity.

WTR/WTRR: The relatedness metric between two tags is denoted by $c_{u_i}(t_x, t_y)$, and represents the degree of correspondence (or connection) between t_x and t_y with respect to user u_i. It measures how similar tag t_y is to a given tag t_x, in the content of a user u_i. The equation to calculate the tag relatedness metric is given by:

$$c_{u_i}(t_x, t_y) = \frac{1}{|M_{u_i,t_x}|} \sum_{m_j \in M_{u_i,t_x}} w_{m_j}(t_y) \tag{4.4}$$

where M_{u_i,t_x} represents the set of movies that user u_i has tagged with tag t_x. We should note that the tag relatedness metric is not symmetric, in the sense that $c_{u_i}(t_x, t_y)$ is not always equal to $c_{u_i}(t_y, t_x)$, as the set M_{u_i,t_x} can be different from the set M_{u_i,t_y}.

For Figure 4.1 we calculate $c_{u_2}(t_2, t_4)$. According to formula we find M_{u_2,t_2} i.e. m_2, m_3. So, $\frac{1}{|M_{u_i,t_x}|} \sum_{m_j \in M_{u_i,t_x}} w_{m_j}(t_y) = \frac{1}{2}\Big(w_{m_2}(t_4) + w_{m_3}(t_4)\Big) = \frac{1}{2}\Big(0 + \frac{5}{2+5+5}\Big) = 0.2083$. Similarly, $c_{u_2}(t_2, t_4) = \frac{1}{2}\Big(w_{m_2}(t_2) + w_{m_3}(t_2)\Big) = \frac{1}{2}\Big(\frac{4}{2+4+4} + \frac{2+5}{2+5+5}\Big) = 0.4916$.

WKRR: In the case of WKRR, the goal is to find the semantic sphere of each keyword for all possible ratings for every users individually. The intuition behind this is that if a user has rated a movie m_i with a high value, the keywords associated to m_i are important to that user. To evaluate the importance of keyword k_x for user u_i who rated m_i with r, we aggregate the movies containing k_x and having rating r together, to form a topic preference.

Since the movie topic can be described by weighted keywords, keywords from all the movies containing k_x and rated with r can be used to describe the topic preference of user u_i. By this, we capture related keywords to calculate the topic preferences. We define $c_{u_i,r}(k_x, k_y)$ to represent the degree of correspondence (or connectivity) between keywords k_x and k_y with respect to user u_i in the context of rating r:

$$c_{u_i,r}(k_x, k_y) = \sum_{m_j \in M_{u_i,k_x}} \frac{1}{|M_{u_i,k_x,r}|} \cdot w_{m_j}(k_y, r) \tag{4.5}$$

where $|M_{u_i,k_x,r}|$ is the number of movies containing keyword k_x that user u_i has rated with rating r.

For Figure 4.3 we calculate $c_{u_2,4}(k_4, k_3)$. According to formula we find M_{u_2,k_4} i.e. m_2, m_3. So, $\frac{1}{|M_{u_2,k_4}|} \sum_{m_j \in M_{u_2,k_4}} w_{m_j}(k_3) = \frac{1}{2}\Big(w_{m_2}(k_3, 4) + w_{m_3}(k_3, 4)\Big) = \frac{1}{2}\Big(\frac{1}{1+1+1} + 0\Big) = 0.1667$. Similarly, $c_{u_2,4}(k_4, k_4) = \frac{1}{2}\Big(w_{m_2}(k_4) + w_{m_3}(k_4)\Big) = \frac{1}{2}\Big(\frac{1}{1+1+1} + 0\Big) = 0.1667$.

Tag/Keyword User Profiles

User Profile Generation From Tags: We are now ready to describe u_i^T in more detail. We start with the original definition as presented by Liang et al. [2010]. Tags, assigned by users, explicitly describe the preferences of the users who assigned them. The number of times a tag is used by a user to describe a movie in the corpus, shows how popular the tag is for that user. Therefore, it becomes necessary to capture the user tag preference (or user tag relevance metric). The value of the user-tag relevance metric signifies how strongly the user feels about a tag:

$$\text{WTR based: } w_{u_i}(t_x) = \frac{n_{u_i,t_x}}{\sum_{t_y \in T_{u_i}} n_{u_i,t_y}} \tag{4.6}$$

where T_{u_i} is the tag set of the user u_i and n_{u_i,t_x} is the number of movies that are collected under the tag t_x by user u_i, in other words, how many times the user has used a certain tag.

For Figure 4.1 we calculate the user-tag relevance metric for user u_2. Similar to movies relevance weight, is estimated for all the tags. User u_2 has used only tag t_2 and tag t_3 for

tagging then $n_{u_2,t_2} = 2$ and $n_{u_2,t_3} = 1$ and $n_{u_2,t_1} = n_{u_2,t_4} = 0$. Hence, $w_{u_2}(t_1) = \frac{0}{1+2} = 0, w_{u_2}(t_2) = \frac{2}{1+2} = .667, w_{u_2}(t_3) = \frac{1}{1+2} = .334$, and $w_{u_2}(t_4) = \frac{0}{1+2} = 0$.

In our vision, u_i^T can be estimated more accurately from ratings. Since our dataset was reduced to a subset of movies for which all users provided both tags and ratings, we are able to refine the above formula, as follows:

$$\text{WTRR based: } w_{u_i}(t_x) = \frac{\sum_{m_j \in M_{u_i,t_x}} r_{u_j,t_x}(m_j)}{\sum_{m_j \in M_{u_i}, t_y \in T_{u_i}} r_{u_i,t_y}(m_j)} \tag{4.7}$$

where the numerator is a summation of the ratings assigned to the movie m_j by all the users who used t_x to annotate it, and the denominator is the summation over all ratings assigned to the movie m_j by all the users who tagged it.

We demonstrate how to calculate the user-tag relevance for the WTRR approach using the example in Figure 4.1.We show the calculation for user u_2. User u_2 uses tag t_2 and t_3 to tag movies. So, $\sum_{m_j \in M_{u_2,t_2}} r_{u_2,t_2}(m_j) = 4 + 2 = 6$ and $\sum_{m_j \in M_{u_2,t_3}} r_{u_2,t_3}(m_j) = 4$ and the rest will be zero. Hence, $w_{u_2}(t_1) = \frac{0}{4+2+4} = 0, w_{u_2}(t_2) = \frac{2+4}{4+2+4} = 0.6, w_{u_2}(t_3) = \frac{4}{4+2+4} = 0.4, w_{u_2}(t_4) = \frac{0}{4+2+4} = 0$.

As tags related to t_y are believed to be representative for user u_i, the weight (relevance) of tag t_y for a user u_i is calculated as summation of relatedness between the tags used by user u_i (i.e., $t_x \in T_{u_i}$) and target tag t_y; $W_{u_i}(t_y)$ is the *total relevance weight* of t_y for the user u_i and is given by:

$$W_{u_i}(t_y) = \sum_{t_x \in T_{u_i}} w_{u_i}(t_x) \cdot c_{u_i}(t_x, t_y) \tag{4.8}$$

For Figure 4.1, based on WTRR values, we build the profile for user u_2 i.e. calculate the weight for all the tags with respect to user. We need to calculate the set of tags user assigned to movies $T_{u_i} = \{t_2, t_3\}$. We show the estimation of $W_{u_2}(t_1)$. As there are two tags that are used by user, we can expand the Equation (4.8) based on user tag set. $W_{u_2}(t_1) = w_{u_2}(t_2) \cdot c_{u_2,t_2}(t_1) + w_{u_2}(t_3) \cdot c_{u_2,t_3}(t_1) = \frac{6}{10} \cdot \frac{0.2+0}{2} + \frac{4}{10} \cdot \frac{0.2}{1} = 0.14$. Similarly, $W_{u_2}(t_2) = 0.45496, W_{u_2}(t_3) = 0.28, W_{u_2}(t_4) = 0.125$.

Similar to inverse document frequency in information retrieval, a tag's occurrence for all users must be taken into consideration in order to measure the *general* importance of a tag in the topic preference identification of a user; $iuf(t_y)$ is the *inverse user frequency* of tag t_y and is given by:

$$iuf(t_y) = \frac{1}{log(e + |U_{t_y}|)} \tag{4.9}$$

where $|U_{t_y}|$ is the number of users that used t_y and e is Euler's number thus, $0 \leq iuf(t_y) \leq 1$.

For Figure 4.1, we calculate the value of $iuf(t_y)$ for all the tags in the corpus. For $iuf(t_1)$ first we find the number of users that used tag t_1 i.e. 1. Hence, $iuf(t_y) = \frac{1}{log(e+1)} = 0.6309$. Similarly, $iuf(t_1) = iuf(t_3) = iuf(t_4) = 0.6309$ and $iuf(t_2) = 0.5$. The tag representation of each user is defined as:

$$u_i^T = \{W_{u_i}(t_y) \cdot iuf(t_y) | t_y \in T\} \tag{4.10}$$

The weights in the tag user profile in Equation (4.14) capture the values of the tag topic preference for each user u_i. For Figure 4.1, u_i^T is given by:

$$\begin{vmatrix} & t_1 & t_2 & t_3 & t_4 \\ u_1 & .. & .. & .. & .. \\ u_2 & 0.14 \cdot 0.63 & 0.45 \cdot 0.5 & 0.28 \cdot 0.63 & 0.125 \cdot 0.63 \end{vmatrix} = \begin{vmatrix} t_1 & t_2 & t_3 & t_4 \\ .. & .. & .. & .. \\ 0.088 & 0.227 & 0.176 & 0.078 \end{vmatrix}$$

User Profile Generation From Keywords: Similar to the tag user profile, we calculate a keyword user profile which captures the keyword topic preference for each user u_i. Here, the goal is to estimate the importance of a keyword to a user, given a particular rating. First, we calculate the user-keyword relevance metric which shows how strong a keyword k_x is relevant to a user u_i when the movies that have k_x as a keyword are rated with r. We have:

$$w_{u_i}(k_x, r) = \frac{n_{u_i,k_x,r}}{\sum_{k_y \in K_{u_i}} n_{u_i,k_y,r}} \tag{4.11}$$

where $n_{u_i,k_x,r}$ is the number of movies that share the keyword k_x and are rated by user u_i with a rating value r, in other words, how many times the user has rated a movie containing keyword k_x with a rating value equal to r.

Figure 4.3, we show the calculation for user-keyword relevance estimation. User u_2 subgraph contains keywords k_2, k_3 and k_4. For $w_{u_2}(k_3, 4)$, we first find $n_{u_2,k_2,4} = 1, n_{u_2,k_2,4} = 1$ and $n_{u_2,k_2,4} = 1$; the rest will be zero for rating 4. Hence, $w_{u_2}(k_1, 4) = \frac{0}{1+1+1} = 0, w_{u_2}(k_2, 4) = \frac{1}{1+1+1} = 0.33, w_{u_2}(k_3, 4) = \frac{1}{1+1+1} = 0.33, w_{u_2}(k_4, 4) = \frac{1}{1+1+1} = 0.33$.

Let K_{u_i} be the keyword set of the user u_i. The total relevance weight of a keyword for a user u_i is given by:

$$W_{u_i}(k_y) = \sum_{k_x \in K_{u_i}, r \in R} w_{u_i}(k_x, r) \cdot c_{u_i,r}(k_x, k_y) \cdot iuf(k_y, r) \tag{4.12}$$

where $iuf(k_x, r)$ is the *inverse user frequency* of keyword k_x rated with a rating r. A keyword's occurrence within movies rated by all users must be taken into consideration in order to measure the *general* importance of a keyword in the topic preference of a user. This is how we calculate the inverse user frequency:

$$iuf(k_y, r) = \frac{1}{log(e + |U_{k_y,r}|)} \tag{4.13}$$

where $|U_{k_y,r}|$ is the number of users that rated movies which contain k_y with a rating value r and e is Euler's number.

Thus, the keyword representation of each user is defined as below:

$$u_i^K(r) = \{W_{u_i,r}(k_y) | k_y \in K\} \tag{4.14}$$

For Figure 4.2, we calculate the value of $iuf(k_y, r)$ for all the possible ratings. For $iuf(k_4, 2)$, we find the number of users that have keywords k_4 with rating 2, i.e. 2. Hence, $iuf(k_4, 2) = \frac{1}{log(e+2)} = 0.5$. For $iuf(k_4, 4)$, there are 2 users that have keyword k_4 with rating 4 thus, $iuf(k_4, 4) = \frac{1}{log(e+2)} = 0.5$.

We calculate $W_{u_2}(k_1)$ based on Figure 4.3. K_{u_2} contains keywords k_2, k_3 and k_4 and although R has 10 distinct values for this particular case, except rating 4 and rating 2, rest can be discarded (as they are not used by user u_2, thus values will add up to zero). We expand $W_{u_2}(k_4) = w_{u_2}(k_2, 2) \cdot c_{u_2,2}(k_2, k_4) \cdot iuf(k_4, 2) + w_{u_2}(k_3, 2) \cdot c_{u_2,2}(k_3, k_4) \cdot iuf(k_4, 2) + w_{u_2}(k_4, 2) \cdot c_{u_2,2}(k_4, k_4) \cdot iuf(k_4, 2) + w_{u_2}(k_2, 4) \cdot c_{u_2,4}(k_2, k_4) \cdot iuf(k_4, 4) + w_{u_2}(k_3, 4) \cdot c_{u_2,4}(k_3, k_4) \cdot$

$iuf(k_4, 4) + w_{u_2}(k_4, 4) \cdot c_{u_2,4}(k_4, k_4) \cdot iuf(k_4, 4) = 0 \cdot \frac{1}{1}\left(\frac{1}{3}\right) \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2}\left(\frac{1}{3} + \frac{1}{2}\right) \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2}\left(\frac{1}{3} + \frac{1}{2}\right) \cdot \frac{1}{2} + \frac{1}{3} \cdot \frac{1}{1}\left(\frac{1}{3}\right) \cdot \frac{1}{2} + \frac{1}{3} \cdot \frac{1}{2}\left(\frac{1}{3} + \frac{1}{2}\right) \cdot \frac{1}{2} + \frac{1}{3} \cdot \frac{1}{2}\left(\frac{1}{3} + \frac{1}{2}\right) \cdot \frac{1}{2} = 0.4027$. Similarly, the rest of the values can be computed.

Movie Preference Profile

As mentioned in Section 4.2, a user profile is two-faceted, comprising of the movie tag/keyword member (u^T/u^K) and of the movie rating based constituent (u^M). Using tags and keywords, we capture the content-based quality of our approach. To acquire the collaborative filtering idea into our system, we use the ratings again, but in a straightforward manner: u^M is a vector with M elements, each corresponding to a movie in the corpus. The values of the elements are either 0 or 1, depending on whether or not the user has rated the movie.

4.3 Neighborhood Formation

In order to predict how much a user will enjoy an unseen movie, in other words to predict their rating for it, we first set out to find the community of users sharing similar taste, aka the N nearest neighbors. The main goal is to identify for each user u, an ordered list of N most similar users, $U = \{u_1, u_2, ..., u_N\}$ such that $u \in U$ and $sim(u, u_1)$ is maximum, $sim(u, u_2)$ is the second highest and so on. The N-nearest users are selected based on the similarity value. The similarity values play a double role in neighborhood-based recommendation methods:

- they allow the selection of trusted neighbors whose ratings are used in the prediction, and
- they provide the means to give more or less importance to these neighbors in the prediction

The selection of the similarity measure is one of the most critical aspects of building a neighborhood-based recommender system, as it can have a significant impact on both its accuracy and its performance. Thus, for computing the similarity between user profiles, the

cosine similarity measure is used (in other words, the similarity between two users measures the cosine of the angle between the profile vectors representing them).

As discussed in Section 4.2.1, each user is encoded with their own topic preferences (captured by tags or keywords) and movie preferences (captured by ratings). The similarity between two users based on user topic preference is denoted as $sim_u^T(u_i, u_j)$ or $sim_u^K(u_i, u_j)$, where T and K are the sets of tags and keywords, respectively.

We use cosine similarity to compute the angle between the two user vectors, which are represented by the set of all tags with weights representing them. Let u_i and u_j be the two weighted tag vector, then $sim_u^T(u_i, u_j)$ is given by:

$$sim_u^T(u_i, u_j) = \frac{\sum_{y=1}^{|T|} u_{i,y}.u_{j,y}}{\sqrt{\Big(\sum_{y=1}^{|T|} {u_{i,y}}^2\Big).\Big(\sum_{y=1}^{|T|} {u_{j,y}}^2\Big)}} \tag{4.15}$$

Similarly, for $sim_u^K(u_i, u_j)$ estimation we use following formula

$$sim_u^K(u_i, u_j) = \frac{\sum_{k=1}^{|K|} u_{i,k}.u_{j,k}}{\sqrt{\Big(\sum_{k=1}^{|K|} {u_{i,k}}^2\Big).\Big(\sum_{k=1}^{|K|} {u_{j,k}}^2\Big)}} \tag{4.16}$$

Whereas, the similarity between two users based on user movie preference is denoted as $sim_u^M(u_i, u_j)$ where M is the set of all movies. User movie preference takes the popularity of movie into the consideration for two users and is given by:

$$sim^M(u_i, u_j) = \frac{\sum_{m_k \in M_{u_i} \cap M_{u_j}} iuf(m_k)}{\sqrt{|M_{u_j}| \cdot |M_{u_j}|}} \tag{4.17}$$

where $|M_{u_i}|$ is number of movies user u_i has tagged, $iuf(m_k)$ is the *inverse user frequency* of movie m_k and is defined as $iuf(m_k) = \frac{1}{log(e+|U_{m_k}|)}$, where $|U_{m_k}|$ is the number of users that have tagged movie m_k.

Given the topic and movie profiles, the similarity between two users is given by:

$$sim(u_i, u_j) = \omega \cdot sim^T(u_i^T, u_j^T) + (1 - \omega) \cdot sim^M(u_i^M, u_j^M) \tag{4.18}$$

in the case of tags, and

$$sim(u_i, u_j) = \omega \cdot sim^T(u_i^K, u_j^K) + (1 - \omega) \cdot sim^M(u_i^M, u_j^M) \tag{4.19}$$

in the case of keywords.

In both cases, ω is a weighting parameter such that $0 \leq \omega \leq 1$. This parameter omega controls the extent of the collaborative dimension of the algorithm. As we decrease the value of ω, the algorithm will be predominantly collaborative, as the contribution of the users movie preferences will dominate. During the experimental phase, we kept omega $\omega = 0.9$.

Since we have the similarities between different users, a set of similar neighbors can be identified. The traditional Top N algorithms choose the Top N most similar neighbors to predict the missing value, which in our case is a prediction for a movie which is not yet watched by a user. Specifically, to predict a missing value $r_{(u,m)}$ in the movie-user matrix, a set of users similar to u, specifically $N(u)$, is denoted by:

$$N(u) = \{v | v \in T(u), u \in U\} \tag{4.20}$$

where $T(u)$ is the set of N most similar users to user u.

4.4 Rating Prediction Formula

To calculate the missing ratings, we used a popular user-based prediction formula described by Herlocker et al. [1999]. The intuition behind this prediction scheme is that user rating distributions spread around different points. For example, one user rates a good movie with 4 and a bad movie with 2, whereas others users are using 1 for bad movies and 3 for good movies. Intuitively, different users judge movies differently, thus user ratings are

inconsistent. This prediction scheme normalizes the rating $r(u_m)$ by dividing the user-mean-centered rating by the standard deviation σ_u of the ratings given by user u.

$$\hat{r}_{(u,m)} = \bar{r}_u + \sigma_u \frac{\sum_{v \in N(u)} \frac{w_{uv}(r_{v,m} - \bar{r}_v)}{\sigma_v}}{\sum_{v \in N(u)} |w_{uv}|} \tag{4.21}$$

Following the notation from [Herlocker et al., 1999], $\bar{r}_u$ is the average of the ratings given by user u, w_{uv} is the similarity value between user u and user v, σ_u is the standard deviation of ratings given by user u and finally, $N(u)$ is set of most similar users to user u.

Chapter 5

Experimental Setup

This chapter presents the experiments conducted to evaluate the performance of our proposed approaches, WTRR and WKRR. First, we provide some statistical information on the dataset that we used. Then, we specify the metrics used for evaluation and we give an overview of how we split the data during the training and testing phases. Finally, we present the experiments designed to investigate the performance of both WTRR and WKRR algorithms with respect to predicting movie ratings.

5.1 Dataset

The data set used in our experiments, hetrec2011-movielens-2k dated May 2011, is made available to the public by Cantador et al. [2011]. It is based on the original MovieLens10M dataset, published by the GroupLens[1] research group. The movies in this data set are also referencing their corresponding web pages at the IMDB website. Information about the format as well as statistics regarding the data are available at the hetrec2011-movielens-2k website.

As shown in Table 5.1, there are 2,113 users, 10,197 movies and a total of 13,222 unique tags that fall into 47,957 tag assignment tuples of the form [user, tag, movie]. There are also 855,598 user ratings ranging from 0.5 to 5.0, in increments of 0.5, thus a total of 10 distinct rating values. There is an average of 405 ratings per user, and 85 per movie. This data set

[1] http://www.grouplens.org

Table 5.1: *Statistics about hetrec2011-movielens-2k and our reduced dataset*

	Movielens Dataset	Reduced Dataset
# users	2113	1097
movies	10197	2655
tags	13222	8288
tag assignment	47899	36855
ratings	855598	762238

has been previously used in [Bothos et al., 2011], [Said et al., 2012] and [Jones et al., 2011].

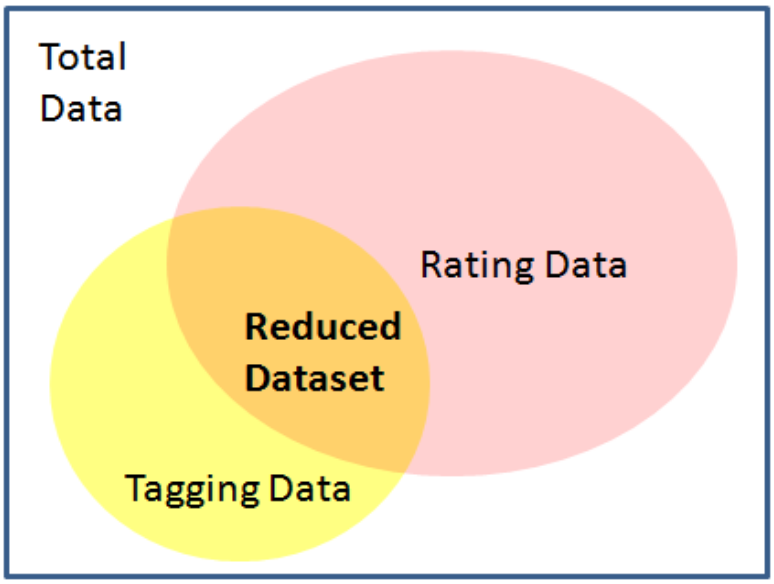

Figure 5.1: *Venn diagram for MovieLens*

Our experimental setup requires some pre-processing of the data. For comparing WTRR and WKRR, we used movies that users have both tagged and rated. Figure 5.1 shows that, after taking the intersection between movies that are rated and movies that are tagged by users, we obtain a subset of 4,655 movies for which every user provided a tag as well as a rating. As shown in Table 5.1, this subset contains 762,238 tag/rating assignments from 1,097 users. A number of 8,288 unique tags were used to make 36885 tag assignments. The density of this dataset is 14.93%, which is satisfying.

Figure 5.2, shows that almost half of the ratings are 4. The mean of the ratings is 3.4379 with a standard deviation of 1.0025. It is also worth mentioning that the hetrec2011-

movielens-2k dataset is relatively dense (3.97 %) as compared to the Netflix Prize dataset whose density is less than 2 %. In addition to the hetrec2011-movielens-2k dataset, for our WKRR aproach we also used content information about the movies from IMDB.

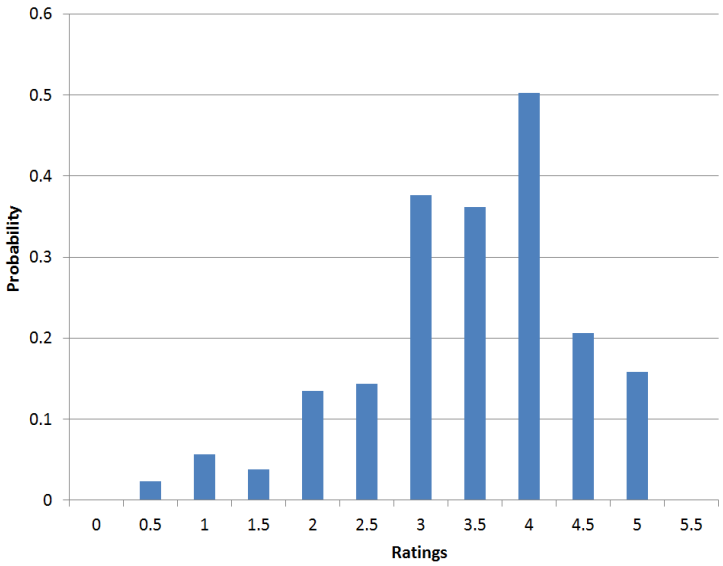

Figure 5.2: *MovieLens rating distribution*

5.2 Evaluation Metrics

Evaluating a recommender system and determining the accuracy of an algorithm is difficult for several reasons. First, algorithms behave differently on different datasets. Second, the goals of the recommenders may vary and evaluation will be based on fulfilling unique criteria. Thus, according to Herlocker et al. [2004], accuracy measures for recommendation systems can be classified into the following categories: predictive accuracy metrics, such as Root Mean Squared Error (RMSE) or Mean Absolute Error (MAE) and its variations; classification accuracy metrics, such as precision, recall, F1-measure and ROC sensitivity; rank accuracy metrics, such as Mean Average Precision (MAP), normalized distance-based performance metric (NDPM), etc.

We can state that simply measuring classification accuracy, the results will be biased for

our dataset, given the skewed distribution of the ratings, as shown in Figure 5.2. As almost half of the ratings are 4 in the dataset and getting a threshold below or above the rating 4 will not serve the purpose, for evaluation of our work, we used two widely forms of error measures. In a survey about collaborative filtering Su and Khoshgoftaar [2009], provided a detailed analysis of evaluation metrics for rating prediction algorithms. The very general form is the average absolute deviation of the predicted rating and actual rating for the movies.

Mean Absolute Error (MAE)

The most widely used metric in CF research literature is Mean Absolute Error (MAE), which computes the average of the absolute difference between the predictions and true ratings.

$$MAE = \frac{\sum_{i,j} |p_{u,i} - r_{u,i}|}{N} \tag{5.1}$$

where N is the total number of ratings over all users, $p_{u,i}$ is the predicted rating for user u on movie i, and $r_{u,i}$ is the actual rating for movie i. The lower the MAE, the better the prediction.

Some studies on recommender system show that MAE is not an effective measure for evaluation when the data is sparse. Thus, we also use Root Mean Squared Error (RMSE) as an accuracy measure for comparing the results of various recommenders with the result of our proposed method. RMSE puts more emphasis on larger absolute errors, and is given by:

$$RMSE = \sqrt{\frac{1}{N} \sum_{u,i} (p_{u,i} - r_{u,i})^2} \tag{5.2}$$

where N is the total number of ratings over all users, $p_{u,i}$ is the predicted rating for user u on movie i, and $r_{u,i}$ is the actual rating for movie i. RMSE amplifies the contributions of the absolute errors between the predictions and the true values.

5.3 Experimental Setup

For evaluation of our proposed algorithms we need to split the original rating data into two sets: a training set and test set. We ensure that these two subsets are disjoint. For the test, we simply ignore 20% of the ratings, and we run our algorithms to measure the deviation from predicted to actual rating value. Using Equation (4.21), we utilize the training set to predict all the hidden ratings in the test set. We use the 5-fold cross validation technique, to eliminate any fortunate or unfortunate cases, while evaluating the performance of our algorithms, so we split the dataset 5 times, into pairs of disjoint training/test subsets. Thus, we use 5 splits of the hetrec2011-movielens-2k dataset and for each split we kept 80% of users in the training set, while 20% of users are in the test set. For each test user, we hide 20% of movie ratings while the remaining 80% of movie ratings are used as training set for a user. We intentionally hide the ratings so that we can objectively evaluate our models.

Given the topic and movie profiles, the similarity between two users is given by:

$$sim(u_i, u_j) = \omega \cdot sim^T(u_i^T, u_j^T) + (1 - \omega) \cdot sim^M(u_i^M, u_j^M) \tag{5.3}$$

in the case of tags, and

$$sim(u_i, u_j) = \omega \cdot sim^T(u_i^K, u_j^K) + (1 - \omega) \cdot sim^M(u_i^M, u_j^M) \tag{5.4}$$

in the case of keywords. In both cases, ω is a weighting parameter such that $0 \leq \omega \leq 1$. This parameter controls the contribution of the collaborative dimension of the algorithm vs. content based recommender. As we decrease the value of ω, the algorithm will be predominantly collaborative, as the contribution of the users movie preferences will dominate. We believe our algorithm can also be helpful in alleviating the problem of cold start. As for new users similarity score from collaborative filtering will be zero or near to zero. Thus, for such user the prediction will solely depends on user topic preference. User topic preference consider the related tags/keywords, so user will have the profile which can be used to predict the rating of unseen movies. We tested the algorithm on various omega ω values, we found

value between 0.85 - 0.90 algorithm perform best. Thus, during the experimental phase, we kept omega $\omega = 0.9$.

5.4 Research Questions

We address the following research questions in this work:

1. How can we use the information represented by IMDB for movies, to support and improve the recommendation performance for a sparse dataset?

2. How can we use the user tags and ratings available in social networking systems to provide more accurate recommendations to users and can we improve performance by combining the ratings with tags?

3. Which result in better performance, keywords or tags?

During our experimentation, we used user-mean center method for prediction computation. We perform 2 experiments in this thesis.

- Experiment performed to compare WKRR (Weighted Keyword based Recommender) with state-of-art methods. WKRR is described in Chapter 4 where we calculated the weight of each keyword (from IMDB) for all the users based on explicit ratings. For neighborhood formation, we applied cosine similarity metrics to calculate the similarity score among users. We used hetrec2011-movielens-2k dataset for experiment.

- Experiment performed to compare WTR (Weighted Tag based Recommender), WTRR (Weighted Tag Rating Recommender) and WKRR (Weighted Keyword Rating Recommender), as described in Chapter 4 where we calculated the weight of each tag or keyword for users based on implicit or explicit ratings. According to the requirements of our approaches, we perform this experiment on reduced hetrec2011-movielens-2k dataset.

5.5 System Architecture for WKRR and WTRR

This section provides an overview of the system architecture of our hybrid recommender systems. Figures 5.4 and 5.3 show the main elements of our design, and also illustrate the workflow of our system.

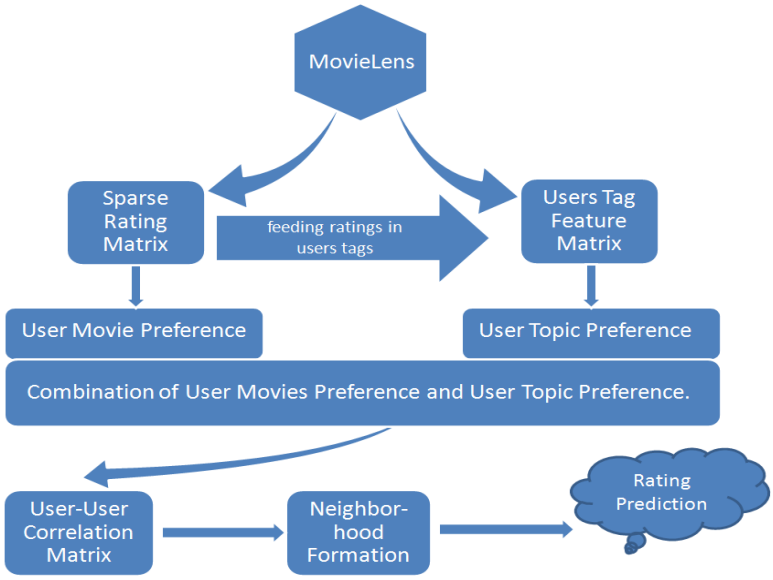

Figure 5.3: *System overview of Weighted Tag Rating Recommender*

Figure 5.3 shows the architecture design of WTRR, which is performed on hetrec2011-movielens-2k reduce dataset. Instead of keywords from IMDB, user-tags are used as features to represent users. Explicit ratings from users are combined with user-tags to form a single weighted user-tag feature matrix. Topic preference for each user is calculated based on association of user-tags and their ratings. WTRR has a user movie preference module, where the movie preference is a binary vector obtained using the tag information. If a user tagged a movie, it is assumed that the user liked the movie; otherwise, the user didn't like the movie. Later, these two matrices are combined linearly and go through similarity estimation process, to determine the neighborhood of the most similar users. Ultimately, to predict the missing ratings for users, user-mean center prediction scheme is used.

One main observation from Figure 5.4 is that the MovieLens and IMDB Keyword data

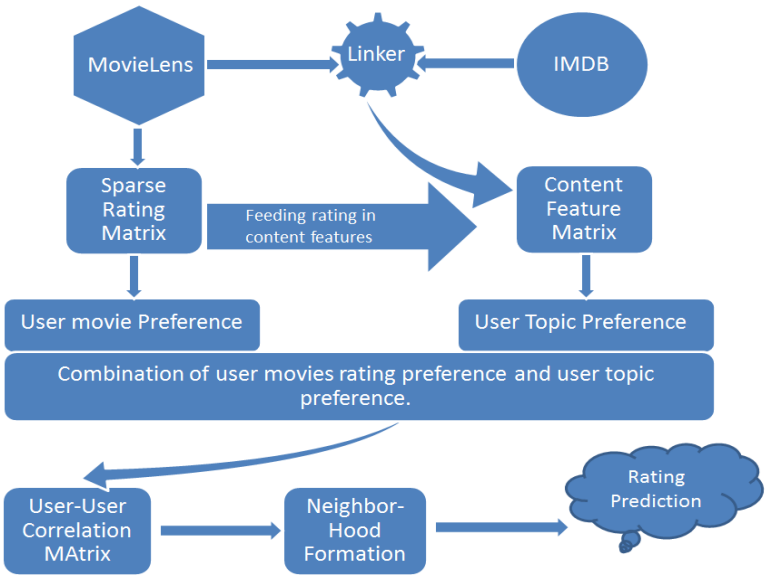

Figure 5.4: *System overview of Weighted Keyword Rating Recommender*

are joined through a Linker written in Java programming language. This linker retrieves the content feature from the IMDB for all the movies in the hetrec2011-movielens-2k dataset and also retains the rating history from the hetrec2011-movielens-2k before they are combined to a single weighted matrix.

Then, based on the content features and all the user ratings, each user topic preference is calculated. We also calculate the user movie rating preference. Then, we linearly combine these two matrices. This extended weighted matrix will go through a similarity calculation process, which transforms this user feature matrix into a user-user matrix. The user-user matrix provides information about the closeness among some subset of users. Finally, we form the neighborhood of the most similar users, based on values from the user-user matrix. Ultimately, we use the user-mean center method to predict missing ratings.

Chapter 6

Results

In this chapter, we discuss the performance of our proposed algorithm based on the experiments described in Section 5.3. The experiments were designed to investigate the effectiveness of using ratings along with tags or keywords versus using just tags when building the preference profiles for the users. To test these approaches, we conducted our experiments on the hetrec2011-movielens-2k dataset. First, we discuss the performances of the WTR, WTRR and WKRR approaches on a subset of the hetrec2011-movielens-2k. Next, we show results for the WKRR when applied to the the whole dataset, in order to compare its performance with other algorithms proposed in the recent literature.

6.1 WTR, WTRR and WKRR

We hypothesized that user profiles based on features captured from movie descriptions, such as tags or keywords, are likely to improve the accuracy of the prediction task when used with explicit ratings, as opposed to using only tags and implicit ratings. To verify this intuition, we ran experiments with WTR, WTRR and WKRR. To objectively compare the results of these approaches, we had to resort to a filtered subset of the hetrec2011-movielens-2k dataset, which had to contain only those users that both tagged and rated the movies they have watched. For the WTR approach [Liang et al., 2010], we only considered the tags assigned by users to the movies. For the WTRR approach, we incorporated tags and ratings, while for WKRR approach we used keywords along with ratings, as described in

Table 6.1: *Performance values in terms of RMSE and MAE, based on experiments run with an increasing number (N) of users in the neighborhood formation.*

N	WTR	WTRR	WKRR	WTR	WTRR	WKRR
	RMSE			**MAE**		
75	0.84601	0.83410	0.84014	0.58827	0.58268	0.58085
50	0.85196	0.83796	0.84533	0.59513	0.57994	0.58210
30	0.86570	0.84821	0.85937	0.60014	0.58461	0.59207
20	0.87815	0.85594	0.87103	0.61129	0.59410	0.60842
10	0.89840	0.87917	0.89005	0.63146	0.61198	0.62372

Section 4.2.

To study the impact of the neighborhood size, we ran experiments with $k \in \{10, 20, 30, 50, 75\}$ for all three approaches and compared the results in terms RMSE and MAE. The values are displayed in Table 6.1. The prediction quality improves with the increase in the neighborhood size for all three cases (WTR, WTRR and WKKR). However, the numbers show a trend of convergence, in the sense that given a larger size of the neighborhood, the predictions are likely to resemble each other.

The values from Table 6.1 show that tags along with implicit ratings are outperformed by the schemes which incorporate the explicit ratings provided by the users. This supports the original intuition about ratings being capable of capturing the real user preferences. The best results on the filtered dataset are obtained with WTRR, and one possible explanation could come from the fact that combining user specific features like the tags, which they personally assign, with user specific ratings, which denote the exact preference, brings more accurate information in the building of the profiles. Thirdly, while keywords are movie dependent rather than user specific, it is not very surprising that WKRR lies in the middle between WTR and WTRR.

6.2 WKRR versus Prior Approaches

For many real world datasets, a lot of movies don't have user tags. This fact makes the tag-based approaches impractical. Instead of using the exact tag, we use keywords, which represent a collective categorization of movies. Results from Table 6.1 reinforce our intuition that assigning different weights to keywords based on explicit rating information is useful. To evaluate the performance of our algorithm we compare it against other recommenders available in recent literature. Therefore, we applied WKRR to the whole hetrec2011-movielens-2k dataset, and ran it with a neighborhood size of 30, to maintain the trend from other papers and do a fair evaluation. We collected previous results reported on the same complete hetrec2011-movielens-2k dataset and show a comparison in Table 6.1. The system proposed by Bothos et al. [2011] is an ensemble of various recommenders, called Information Market Based recommender Fusion (IMBrf), primarily used for mining and aggregating the information from various sources. This technique is inspired from the market, where information from heterogeneous sources is incorporated to make predictions about future events. We compare our results with the results of other approaches as reported by Bothos et al. [2011], including a pure collaborative filtering technique (CF) and a content-based recommender system, called content analysis (CA). For CF, the authors used the neighborhood based approach and set the size of the neighborhood to 30. We will preserve these settings in our experimental setup, to be able to make realistic comparisons. The CA recommender is based on latent topic analysis, and movies are mapped to topics via tags. The prediction is made by finding topics in new movies that are correlated to the user profiles. Another recommender based on averaging the ratings (AVGR) is presented in [Bothos et al., 2011], where an unrated item's rating is estimated from the weighted average of other ratings from other users. Finally, Linear Least Square (LLS) is proposed in [Bothos et al., 2011]. LLS is a linear combination of CF, CA and AVGR: $R_{LS} = \alpha CF + \beta CA + \gamma AVGR$, where R_{LS} denotes the predicted rating (the parameters α, β and γ are found by optimization, for more details see [Bothos et al., 2011]). In [Jones et al., 2011], the authors propose learning

Table 6.2: *Performance of our proposed approach, WKRR, compared with other results from recent work reported in the literature, on the complete hetrec2011-movielens-2k dataset. Performances for IMBrf, pure CF, CA, LLS and AVGR were obtained from [Bothos et al., 2011], while the PMF value is from [Jones et al., 2011].*

WKRR	IMBrf	CF	CA	AVGR	LLS	PMF
0.8304	0.8797	0.8876	0.9436	1.088	0.8758	0.8367

multiple models which can incorporate different types of inputs to predict the preferences of diverse users. Probabilistic Matrix Factorization (PMF) is a variational Bayesian inference technique, used to alleviate the overfitting problem in singular value decomposition (SVD) approaches. Priors are introduced and parameters are estimated using variational Bayesian inference [Nakajima and Sugiyama, 2007]. PMF models the user preference matrix as a product between the lower-rank user and movie matrices [Jones et al., 2011].

Based on the results from Table 6.2, we can conclude that WKRR outperforms other approaches in the neighborhood category and is very useful in capturing the user preferences from relatively large datasets. WKRR features are generated from the movies descriptions along with users ratings when the user profiling is done, and they outnumber the features available to the WTR strategy, which uses just tags. WKRR allows us to use the hetrec2011-movielens-2k dataset in its entirety. We observed from Table 6.2 that the movie keyword analysis is, in this context, more effective than the user tag analysis. One possible explanation for such behavior might be that there is a difference in the sparsity levels of the two feature spaces.

Chapter 7

Conclusions and Future Work

In this chapter, we will discuss the questions raised in this thesis, using the results reported in Chapter 6. Some improvements and new directions for this work are proposed in Section 7.2.

7.1 Conclusions

In this work, we propose a novel hybrid recommendation technique WKRR (Weighted Keyword Rating Recommender) and WTRR (Weighted Tag Rating Recommender) for combining the collaborative filtering and the content based recommendation algorithms and show that WTRR/WKRR algorithms outperform other approaches in the neighborhood category. Our proposed WKRR was built using the hetrec2011-movielens-2k dataset, supplemented with extra movie information from the IMDB online archive. We alleviate the noise and synonymy of keywords by considering a pool of related terms when constructing the profiles. We also modified the WTR (Weighted Tag Recommender) by Liang et al. [2010] to leverage the rating information from the users along with tags. WThis approach, called Weighted Tag Rating Recommender (WTRR), uses users explicit ratings to calculate the users topic preferences. Problem of synonymy and tag ambiguity is addressed by considering the pool of related tags. Secondly, to use the collaborative based filtering, user-user similarities are calculated for both the approaches. Using the similarity estimation approach, k-nearest neighbors are found and based on their relatedness, we determine ratings for unseen movies.

The experiments described in Section 5.3 are designed to answer the research questions

raised in Section 5.4. We briefly explain them in the following paragraphs. The results of our experiments show that the performance of WKRR exceeds the other approaches. Our results demonstrate that for datasets in which relevant tag information is scarce, extending the features from tags to movie keywords and ratings boosts the performance. Explicit ratings improve the recommendations over the implicit ratings (inferred from the presence/absence of tags) as well. Since tags and ratings are user-specific, it is not surprising that using them improves the recommender. The quality of the training data is crucial in building any information filtering frameworks. Between keywords (representing the collective user information) and tags (which are specific to each user) we observed that both give comparable results, with a slight increase in the tag-based approach.

The idea for the work presented in this thesis originates from [Liang et al., 2010]. However, our adaptation and proposed approach are applicable to any domain that has item descriptions and users willing to rate the items.

7.2 Future Work

As a next step into our future research, we will start by tweaking the algorithm we proposed. For example, we could tune the user movie preference profile by incorporating actual rating values into the representation of the u^M vector. (See Section 4.2.1). This modification would also help us use other similarity measurements, such as Pearson Correlation Coefficient.

This work focuses on profiling each user based on ratings, but we can also profile movies in same feature space so that similarity between user and movies can be determined. Applying an improved customized prediction scheme would also be a good addition. Using our approach on clusters of users with similar taste may also increase the relevance of the predictions.

A dataset close to 10M ratings is 10% of what present recommendation systems are required to handle (for example hetrec2011-movielens-2k). In future work, we plan on

using larger versions of the MovieLens[1] datasets, for example MovieLens1M, consisting of 1 million ratings from 6000 users on 4000 movies, or even MovieLens10M which comprises 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. Therefore, the amount of time required for processing such a large dataset is considerably greater. WKRR/WTRR have high time complexity and even though most of the processing can be done offline, calculating weights for a huge feature space is expensive. A time-saving implementation would be to port our hybrid approach in a Map-Reduce framework. Another avenue we can consider would be to reduce the dimension of matrices by means of singular value decomposition. Moreover, it would be interesting to study another dataset from a different domain.

[1] http://www.grouplens.org/node/12

Bibliography

G. Adomavicius and A. Tuzhilin. Toward 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):734–749, 2005.

X. Amatriain, J. Pujol, and N. Oliver. I like it... i like it not: Evaluating user ratings noise in recommender systems. In *User Modeling, Adaptation, and Personalization*, Lecture Notes in Computer Science. 2009.

M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

R.M. Bell, Y. Koren, and C. Volinsky. The bellkor 2008 solution to the netflix prize. 2008.

E. Bothos, K. Christidis, D. Apostolou, and G. Mentzas. Information market based recommender systems fusion. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pages 1–8. ACM, 2011.

J.S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12:331–370, November 2002.

I. Cantador, P. Brusilovsky, and T. Kuflik. Second workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the fifth ACM conference on Recommender systems*, pages 387–388. ACM, 2011.

M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *In Proceedings of ACM SIGIR Workshop on Recommender Systems*, 1999.

M. Clements, A.P. De Vries, and M.J.T. Reinders. The task-dependent effect of tags and ratings on social media access. *ACM Trans. Inf. Syst.*, 28(4):21:1–21:42, November 2010.

M. De Gemmis, P. Lops, G. Semeraro, and P. Basile. Integrating tags in a semantic content-based recommender. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 163–170. ACM, 2008.

S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli. The adaptive web. chapter User profiles for personalized information access, pages 54–89. Springer-Verlag, Berlin, Heidelberg, 2007.

J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 230–237, New York, NY, USA, 1999. ACM.

J.L. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, January 2004.

S. Huang. Comparision of utility-based recommendation methods. 2008.

C. Jones, J. Ghosh, and A. Sharma. Learning multiple models for exploiting predictive heterogeneity in recommender systems. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, HetRec '11, pages 17–24, New York, NY, USA, 2011. ACM.

V. Kostakos. Is the crowds wisdom biased? a quantitative analysis of three online communities. IEEE Computer Society, 2009.

H. Liang, Y. Xu, Y. Li, R. Nayak, and G. Shaw. A hybrid recommender systems based on weighted tags. May 2010.

Schmidt-Thieme L. Jaeschke R. Hotho A. Stumme G. Symeonidis P. Marinho L.B., Nanopoulos A. Social tagging recommender systems. In Shapira B. Kantor P. Ricci F., Rokach L., editor, *Recommender Systems Handbook*, pages 615–644. Springer, 2011.

B.N. Miller, J.A. Konstan, and J. Riedl. Pocketlens: Toward a personal recommender system. *ACM Transactions on Information Systems (TOIS)*, 22(3):437–476, 2004.

T. Miller. Drowning in information and starving for knowledge. *Journal of Communication*, 1:123–135, 2007.

B. Mobasher, R. Burke, R. Bhaumik, and C. Williams. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Technol.*, 7(4), October 2007.

S. Nakajima and M. Sugiyama. Implicit regularization in variational bayesian matrix factorization, 2007.

K. Papineni. Why inverse document frequency? In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA, 2001. Association for Computational Linguistics.

M. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13:393–408, 1999. 10.1023/A:1006544522159.

M. Pazzani and D. Billsus. Learning and revising user profiles: The identification of interesting web sites. *Machine learning*, 27(3):313–331, 1997.

P. Resnick and H.R. Varian. Recommender systems. *Commun. ACM*, 40(3):56–58, March 1997.

J.J. Rocchio. Relevance feedback in information retrieval. 1971.

A. Said, E.W. De Luca, B. Kille, B. Jain, I. Micus, and S. Albayrak. Kmule: a framework for user-based comparison of recommender algorithms. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*. ACM, 2012.

B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

J. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative filtering recommender systems. *The adaptive web*, pages 291–324, 2007.

J. Ben Schafer, Joseph Konstan, and John Riedi. Recommender systems in e-commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*, EC '99, pages 158–166, New York, NY, USA, 1999. ACM.

G. Shani, R.I. Brafman, and D. Heckerman. An mdp-based recommender system. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 453–460. Morgan Kaufmann Publishers Inc., 2002.

U. Shardanand and P. Maes. Social information filtering: algorithms for automating word of mouth. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217. ACM Press/Addison-Wesley Publishing Co., 1995.

X. Su and T.M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:4:2–4:2, January 2009.

J. Surowiecki and M.P. Silverman. The wisdom of crowds. *American Journal of Physics*, 2007.

L.H. Ungar and D.P. Foster. Clustering methods for collaborative filtering. In *AAAI Workshop on Recommendation Systems*, number 1. AAAI Press, 1998.

R. Van Meteren and M. Van Someren. Using content-based filtering for recommendation. *Structure*, 184(1):47–56, 2000.

R.R. Walia. Collaborative filtering: A comparison of graph-based semi-supervised learning methods and memory-based methods. *Strengthening the Role of ICT in Development*, page 70, 2008.

Z. Wang, Y. Wang, and H. Wu. Tags meet ratings: Improving collaborative filtering with tag-based neighborhood method. In *Proceedings of the Workshop on Social Recommender Systems (SRS10), Hong Kong, China*, 2010.

T. Yoneya and H. Mamitsuka. Pure: a pubmed article recommendation system based on content-based filtering. In *Genome Inform*, volume 18, pages 267–276, 2007.