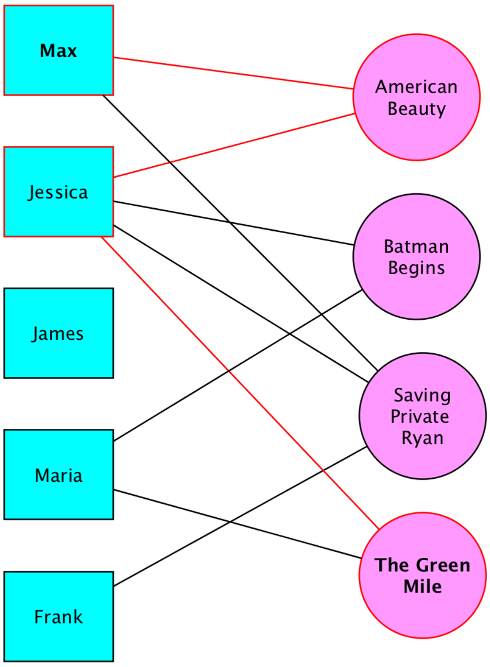
University of Zurich[UZH]

Modeling of User Preferences using graph-based Recommender Systems



Thesis October 21, 2015

Thomas Brenner
of Kreuzlingen TG, Switzerland

Student-ID: 08-928-434
thomas.brenner@access.uzh.ch

Advisor: **Bibek Paudel**

Prof. Abraham Bernstein, PhD
Institut für Informatik
Universität Zürich
http://www.ifi.uzh.ch/ddis

Acknowledgements

First, I would like to thank Prof. Dr. Abraham Bernstein for giving me the opportunity to write my master's thesis at the Dynamic and Distributed Information System Group of the University of Zurich. I must also express my greatest appreciation to Bibek Paudel, my supervisor, who helped and guided me throughout the entire work. His contributions to this thesis were manifold, especially new ideas, explanations, encouragement and critique. And last, I am very grateful for the support of my parents, my brother and my friends throughout my studies.

Zusammenfassung

Recommender Systeme sind ein wichtiges Hilfsmittel, um der immensen Flut von Information aus dem Internet Herr zu werden. In den letzten Jahren hat sich der Fokus verlagert. Während zuvor nur die Verbesserung von Genauigkeit im Zentrum stand, rückt nun die Verbesserung der Benutzerzufriedenheit vermehrt ins Zentrum. Diese Arbeit versucht tiefere Kenntnis über Diversität und wie Benutzer darauf reagieren zu gewinnen. Benutzer werden in zwei Gruppen eingeteilt: eine Gruppe, welche Diversität sucht, und eine andere, welche weniger Tendenz zu Diversität zeigt. In dieser Arbeit werden verschiedene Methoden, um diese Gruppen zu bilden ausgearbeitet. In einem zweiten Teil werden Veränderungen an Graphen-basierten Recommender Systemen, im Detail das Anwenden des *tf-idf* Schemas oder das Nutzen von Nachbarschaftsbeziehungen zwischen Benutzern, diskutiert. Dieses Separieren von Benutzern in verschiedene Gruppen und die Variationen an Recommender Systemen werden evaluiert und eine geeignete Kombination wird vorgeschlagen, um die Präferenzen der Benutzer bestmöglich abzubilden. Diese neuen Varianten von Recommender Systemen liefern genauere Vorschläge und gleichzeitig Vorschläge mit höherer Diversität für einige Benutzergruppen verglichen mit modernen Recommender Systemen.

Abstract

Recommender systems have become an important tool to help conquer the immense flood of Internet information. In recent years, focus has shifted from just increasing accuracy to improving user satisfaction by producing more diverse recommendations. This thesis seeks deeper knowledge about diversity and how a user approaches it. Users are assigned to two different groups: a diversity-seeking and non-diversity-seeking group; this paper explains different ways to separate the groups. In a second part, alterations to graph-based recommender systems, i.e. applying the *tf-idf* scheme and employing users' neighborhood relations are discussed. Separation of users into different groups and recommender system variations are evaluated; a useful combination to optimize the results according to a user's preferences is proposed. These new variations of recommender systems succeed in providing more accurate and at the same time more diverse recommendations for certain groups of users compared to state-of-the-art recommender systems.

Table of Contents

1

Introduction

In the current age of information overload, a real challenge is to find any kind of relevant content [Adomavicius and Kwon, 2012]. Brynjolfsson et al. [2006] stated that, in earlier times, all areas of industry were eager to introduce only selected products and services expected to become blockbuster hits in the mass market, ignoring niche products. The term *long-tail*[1] was introduced in this context to illustrate that only a few products service the largest part of the market, while most products are not often sold or consumed. Companies concentrated mainly on producing these potential blockbuster hits. Over the last several decades, digitalization and the Internet revolutionized many of these industries and led to complete new market structures; a vast variety of products can now be produced and purchased easily, making products of the *long-tail* interesting [Brynjolfsson et al., 2006]. The new array of products allows consumers to obtain more ideal, customized and individual products. This possibility in some of the most important industries, like music, movies and books, led to a cultural shift from hit products to niche products [Fleder and Hosanagar, 2009].

A problem resulting from this trend is the development of new techniques and strategies to find the ideal product [Fleder and Hosanagar, 2009]. To cope with this situation and help users find the desired content, new tools were created, like search engines and recommender systems. Whereas search engines help a user find what he seeks, mainly focusing on accuracy, recommender systems offer the user the possibility to reach new content. Recommender systems help users find relevant, accurate and personalized content and have become an important and popular topic in computer science lately because of their great influence on users' behavior[2].

Over time, different methodologies were developed for implementing recommender systems. These recommender systems have different strengths and weaknesses: e.g. one recommender system has a high computation time for large datasets, while another lacks the necessary expected accuracy. These features must be analyzed to optimize recommender systems's usage and achieve the highest possible user satisfaction. The majority of recommender system literature focuses on increasing recommendation accuracy. In some of these rather traditional systems, other aspects of what a user may expect are neglected, like a certain level of surprisal, diversity and the ability to propose content that

[1] *long-tail* is a statistical distribution, meaning that a few elements occur often, while the majority of elements occur only a few times.

[2] Pariser [2011] stated that Amazon generates 35% of the sales using recommender systems.

can only be found in the long-tail [Rajaraman and Ullman, 2011]. It has been shown that recommender systems narrow the content diversity of a specific user over time [Nguyen et al., 2014]. This thesis tries to minimize the narrowing effect of recommender system and help users searching for more diversity. One of the underlying assumptions of this thesis is that, according to Ziegler et al. [2005], user satisfaction can be increased by more diverse recommendations. Traditionally, in recommender systems, accuracy and diversity can be altered only within the algorithm itself. This thesis approaches the process of bringing adequate diversity recommendations to users in a new way. Prior to the evaluation with a recommender system, the first few ratings a user conducted are analyzed to classify his general tendency towards diversity. After classifying a user, the recommender system is adjusted according to his diversity-seeking tendency. The base for the recommender systems builds the vertex transition probabilities of random walks of length three. Based on this recommender system, improvements to introduce a more appropriate amount of diversity are developed. A central reason for finding more diversity is to analyze a user's neighborhood. Other recommenders are also introduced to consider different aspects, find the optimal solution and maximize user satisfaction.

Chapter 2 gives insights into the foundations of recommender systems and current research in the environment of recommender system. Chapter 3 analyzes the dataset and introduces different algorithms to separate users into a diversity-seeking and non-diversity-seeking group. Chapter 4 explains the functionality of a graph-based random walk and introduces alternative solutions to produce higher user satisfaction for diversity-seeking users. In Chapter 5, these new techniques are compared with more traditional recommender systems using well-elaborated metrics measuring accuracy and diversity. Conclusions and findings are discussed in Section 6, followed by limitations and future work in Chapter 7.

1.1 Motivation and Research Question

This section explains the motivation behind the ideas on improving recommender systems and user satisfaction. Recommender system may serve multiple economic purposes, including business models (e.g. Spotify, Netflix) where it is essential to service in a *long-tail* because of items' licensing fees [Brynjolfsson et al., 2006]. A better understanding of diversity and how it is distributed among the users can be essential to create better specifications and to provide users not only with the recommendation they want, but also with the recommender system best tailored to their profile.

While normally, only the recommender itself is improved, this thesis tries a different approach. The process of recommending is split into two parts: a separation of users prior to the recommendation process and evaluation of different existing and new recommender system based on a random walk, with the separated groups of users. The goals can be summarized with the following research questions:

- **Research Question 1:** Does the separation of users into different levels of diversity-seeking have an impact on recommender performance metrics? How can this effect be characterized?

- **Research Question 2:** Can the separation of users according to their diversity-seeking tendency help to improve the modeling of user preferences in recommender systems? What do these modified recommender systems based on the separation of users according to their diversity preference look like?

The two research questions are answered by a detailed evaluation of different graph-based recommender systems with different approaches to categorizing users according to their diversity-seeking tendency. The evaluation concentrates on identifying strengths and weaknesses of the proposed separation algorithms and modified recommender systems.

Separation of users into different categories was influenced by Nguyen et al. [2014]. Their work showed that users behave differently after using recommender systems. They devise a methodology to separate users into a recommendation-following and a recommendation-ignoring group to measure this effect. This was done by analyzing their preferred choices before and after first recommendations. User diversity-seeking tendency has not been studied in existing research. It may be possible that users, similar to the work of Nguyen et al. [2014], have different preferences in their tendency for diversity and recommender systems can be improved by further knowledge about this tendency.

To perform the actual recommendation, a recommender system has to be used as a basis, to adjust it to the specific need of the two groups. This thesis uses a graph-based recommender system as the basis recommender systems. Christoffel [2014] summarized the many advantages of graph-based recommenders, e.g. the fact that graph-based recommenders overcome the limited coverage problem [Desrosiers and Karypis, 2011], the high flexibility of graphs to incorporate meta data [Lee et al., 2012], or its scalability [Gori et al., 2007]. These reasons, along with the possibility to be easily able to modify different aspects of the algorithm, were among the motivations for using a graph-based recommender system to increase diversity.

1.2 Contributions

Contributions of this work include: (i) introduction of the concept of analyzing users according to their diversity preferences, (ii) different separation algorithms (Section 3.4) according to different metrics (Section 3.3) that analyze this preference, (iii) alterations of graph-based algorithms to improve their ability to adapt better to more and less diversity-seeking users and (iv) evaluation of the separation algorithm and the modified random walk algorithms, according to accuracy and diversity metrics. (ii) and (iii) made it necessary to develop new algorithms. Three new algorithms are presented to find diversity-seeking users. In Chapter 4, two new variants of graph-based random walk algorithms are developed with the assumption that diversity-seeking users do indeed prefer diversity.

Separation of diversity-seeking and non diversity-seeking users had a strong influence on the results. The goal was to achieve higher diversity scores than the baseline algorithms for the diversity-seeking users and, at the same time, increase accuracy for non-diversity-seeking users. The results of the evaluation (Chapter 5) could not verify this goal for all proposed algorithms. Separation of users by their affinity for diverse movie choices led to other interesting results. User separation provided a strong increase in both accuracy and diversity for the non-diversity-seeking users.

The chosen approach allowed the combination of collaborative filtering (via graph-based algorithms) and content-based information (separation according to content information). This work proposes a hybrid recommender system that combines meta and feedback information in order to maximize the user satisfaction. The results show that by separating users and by a useful combinations of different recommender systems existing metrics both accuracy and diversity can be increased.

1.3 System Setup

The implementation of this approach is split into two parts; first, separation of users into diversity-seeking and non-diversity-seeking groups and the actual recommender system were conducted in a stand-alone Python project. Python was chosen because of its high readability and its capability to express concepts in fewer lines of code. The Python code produced text files with recommendations. To evaluate the recommendations and to compare them with baseline recommenders, the modified version of MyMediaLite by Christoffel [2014] was used. MyMediaLite is a lightweight, multi-purpose library of recommender systems [Gantner et al., 2011]. This framework was extended by an interface to read in the recommendations lists produced by the Python code. The extended MyMediaLite library was useful because of the many metrics (accuracy and diversity) already implemented and the possibility to compare results with in this framework implemented baseline recommenders.

All calculations were performed on local commodity hardware, necessitating some limitations; size of the graph and the number of iterations steps were limiting factors. Due to calculations with a larger number of ratings, a dataset with not more than 1'000'000 ratings (6'040 users and 3'646 items) was chosen. The number of iterations of one random walk was a trade-off between computational time and precision. Christoffel et al. [2015] stated that, above 15'000 random walks, the results strongly converge; thus, this number of iterations was chosen for every user.

In summary, the system was designed to be as simple as possible and to allow the most possible influence on all parts of the system. The elements to evaluate the performance and the comparison with baseline recommenders were done by the MyMediaLite library.

2

Foundations and Related Work

This chapter discusses several important foundations of recommender systems and summarizes influential research, as well as giving an overview about current problems in the recommender system environment.

2.1 Foundations

This section explains certain important details of and foundations underlying recommender systems. The basis of all recommender systems is a utility matrix [Rajaraman and Ullman, 2011]: a matrix of size $F = |U| \times |I|$, where U is the is the set of all users and I the set of all items. If a user u rated an item i, $F_{u,i} \neq 0$. This value $\neq 0$ represents known preferences of this specific user u for item i. It is important to know that the utility matrix is, in all likelihood, sparse. If $F_{u,i} = 0$, no explicit information exists about the preference of u towards item i.

In general, there are two kinds of recommender systems basing recommendations on alternative foundations: content-based and collaborative filtering recommender systems. These two can be described as follows:

- **Content-based Filtering:** Recommendations are based on item characterization. For example, if a user watched several science-fiction movies, other science-fiction selections are recommended. [Rajaraman and Ullman, 2011]

- **Collaborative Filtering:** This class of recommender systems is based on a similarity measure between users and the items. As opposed to the content-based recommender, this class recommends items that were already purchased, or rated, by similar users. To find similar users, a distance measure is calculated, that is minimized by different techniques and strategies to find the most similar users and recommend the same items they ordered. [Rajaraman and Ullman, 2011]

Content-based recommender systems do not rely on ratings, unlike collaborative filtering that depends heavily on previous object ratings. In the last few years, collaborative filtering has become more popular than content-based filtering. Ricci et al. [2011] introduced the term hybrid recommender system (HRS), which is a combination of both techniques. HRS aims to compensate for the disadvantages of one group – e.g., the problem with collaborative filtering is the difficulty of recommending new items (discussed in Section 2.2) – and retain collaborative filtering advantages. The disadvantages of recommending new or diverse items can be fixed by a more content-based approach; this technique makes recommendations based on content, which is more readily available than ratings. The disadvantages of strictly content-based recommender systems is discussed by Desrosiers and Karypis [2011]. They defined the term *over-specialization* to address the problem that only very similar items can be recommended, because the systems fail to detect items that are different, but liked by the same users.

The approach developed in this thesis is a form of HRS. Its basis is a recommender that is exclusively a collaborative filtering technique. This approach is enhanced by analyzing content of items a user has chosen, to define a similarity between two users based on the rated contents.

For the collaborative filtering recommender systems, two different approaches can be defined: an item-based or user-based algorithm [Sarwar et al., 2001]. User-based algorithms use all relationships between users and items to find a user neighborhood u (e.g. rating the same movie as another user). Once a neighborhood is found, recommendations are built based on neighbors' preferences. User-based collaborative filtering is a popular form in recommender systems and widely used [Sarwar et al., 2001]. On the other hand, the item-based collaborative filtering technique produces a model of user ratings. These models compute the expected value of a user prediction, given users' ratings on other items. There are many *machine learning* algorithms: for example, the Bayesian network. In contrast to the user-based approach, this methodology applies algorithms that calculate the association between two items and build recommendations based on the strength of these associations. [Sarwar et al., 2001]

This section introduced recommender systems by giving an overview about currently important streams in their environment, while the thesis overall focused on graph-based algorithms in collaborative filtering, enhanced by specific elements of the content-based approach. In Section 2.2, literature and current related work in the recommender system environment are summarized.

2.2 Related Work

This section summarizes research on topics discussed in this work. Important subjects addressed are: application of graph theory to recommender system problems in the past, importance of *diversity* and *serendipity* for the user satisfaction and how this problem was tackled previously, the important topic of *cold-start* problems and the narrowing effect of recommendations over a longer period.

The first research using random walks as collaborative filtering was done by Fouss et al. [2005]. In this paper, the random walks on graph were interpreted as Markov chains. With the help of these chains, a procedure was proposed to calculate dissimilarities between nodes of an undirected graph. This method was applied on a Multi Agent System, where each user is an agent, each item an agent and interaction is a link between the two agents. The model produced transition probabilities to the links (edges) between the agents (nodes). Fouss et al. [2005] suggested two quantities to make a ranked list of vertices by their similarity from a starting node. The two quantities are: Average Commute Time and Average First-Passage Time. The first quantity counts the average number of steps a user has to take before reaching a specific item. The second quantity is a distance measure between two nodes in the graph. An item is recommended if one, or both, of the quantities is short, on average. A third quantity worth mentioning is the pseudoinverse of the Laplacian matrix, which is a measure for the similarity of two nodes in the graph. Fouss et al. [2005] stated that the approach using the pseudoinverse of the Laplacian matrix as quantity outperforms the other two quantities. All of these three quantities represent key elements of graph-based recommender systems and are noteworthy.

The random walk approaches used in this thesis rely on the work of Cooper et al. [2014]. Their work differentiated between two approaches: calculation of the transition probabilities using matrix algebra, or performing simulations to estimate transition probabilities. The basis of their random walks are undirected bipartite graphs consisting of users and items. With a sufficient number of walks, simulation results converge with the results calculated by matrix algebra. Cooper et al. [2014] achieved better accuracy than the algorithm proposed by Fouss et al. [2005].

After the wide application of graph-based recommender systems, and collaborative filtering algorithms in general, other problems occurred. One important topic, as already stated in Section 2.1, is the famous *cold-start* problem. Recommender systems using collaborative filtering rely on sufficient previous data (e.g. ratings or purchases). With new or niche products, these products or items are not yet rated and a recommendation can be difficult with collaborative filtering. This problem is known as *cold-start* problem and the long-tail problem [Park and Tuzhilin, 2008]. Schein et al. [2002] also discussed possibilities for recommending items in a previously unrated set. The authors solved the *cold-start* problem by including content-based information to improve the cold-start of their collaborative filtering algorithm. Park and Tuzhilin [2008] proposed a different strategy to deal with the *cold-start* problem. They suggested a separation between the head and tail parts of the items, where only the tail part is clustered. The recommendations for items in the tail are based on ratings in these clusters. This approach, if indeed a proper clustering can be conducted, recommendation error rates for tail items can be reduced. Park and Tuzhilin [2008] used nine different predictive models and two error measures (RSME, MAE) to prove the decreasing error in the long tail. Additionally, with total clustering, a methodology is introduced to cluster the data set, asserting that clustering can not only maintain performance level, but can even improve it.

Another widely discussed topic in recommender systems is the importance of recommendation diversity. According to Ziegler et al. [2005], topic diversification in recommender systems was important to improve user satisfaction. Ziegler et al. [2005] stated that, for example, Amazon has mainly similar items in their recommendations, thus diminishing user satisfaction. Numerous large-scale online and offline evaluations find that user satisfaction is not equal to accuracy in recommender systems. One approach involves balancing top-N recommendation lists with the goal of capturing the full range of the active user's interest. Additionally, a new intra-list similarity metric is being introduced to capture list diversity.

Adomavicius and Kwon [2012] and Herlocker et al. [2004] also stressed the importance of diversity; the term can be more broadly defined by 'nearby' and similar terms like serendipity and novelty, which have an influence on user satisfaction. Herlocker et al. [2004] even stated that accuracy, per se, is useless for practical purposes. Adomavicius and Kwon [2012] identified item popularity as a key factor in diversity reduction and proposed a recommender system with an integrated item-popularity-based raking that increases the probability of recommending a less popular item. McNee et al. [2006] pointed out that accuracy metrics can diminish the relevance of a recommender system and may induce a user to leave the recommender. A well-known phenomena is that if a user rated/purchased a *Harry Potter* book, an accurate recommender may propose the next book in the series; but obviously, in this example, the user is already aware of the next books in the series. A more diverse recommendation, i.e. a recommendation with serendipity, would be much more useful in this scenario.

Zhou et al. [2010] proposed a hybrid recommenders system based on vertex ranking algorithms. Along with traditional accuracy metrics, it is interesting to note that *Personalization* and *Novelty/Surprisal* are important as well, proving that a useful combination of accuracy-featuring and diversity-featuring algorithms can improve user satisfaction.

Another approach to increase user satisfaction by combining accuracy and diversity was proposed by Zhang et al. [2012]. In their work, a framework with the name *Auralist* was introduced to increase the four main aspects of a recommender system simultaneously: *accuracy*, *diversity*, *novelty* and *serendipity.* The goal of *Auralist* was to mimic the actions of a trusted friend, an expert and, at the same time, provide a personalized list of recommendations. Again the authors introduced some non-accuracy-based metrics to capture the effect of providing users with a greater diversity. Zhang et al. [2012] proposed two algorithms: the *Basic Auralist*, that clusters data according to their distance with an approach similar to a word count algorithm in a document. Or, *Bubble Aware Auralist* made a bubble for the user to find a recommended object outside of the cluster (declustering). The authors suggested combining these two algorithms in a hybrid version. It has been shown that, in fact, it is possible to increase the serendipity at the costs of accuracy, but even with reduced accuracy, participants expressed satisfaction with the serendipitous recommendations. This paper supports the Ziegler et al. [2005] premise that users are, indeed, willing to sacrifice accuracy for more diversity/novelty/serendipity. Diversity/novelty/serendipity can be improved without any trade-off between them, except accuracy.

Along with the *cold-start* problem, another phenomenon occurred in recommender systems: positive cycles. Pariser [2011] researched whether recommender systems expose users to narrower content over time. They investigated the experience of users who took recommendations other than that of users who do not regularly take recommendations. The problem of narrowing down and the resulting positive cycles (the rich become even richer) is a widely discussed topic in the recommender system environment. Pariser [2011] separated users into two groups: an 'ignoring' group, that does not follow recommendations and a second, following group that does listen to recommendations. Conclusion: it is a natural thing that preferences are solidified, because habits are established based on what was consumed recently. The goal of the paper was to understand the broadening or narrowing influence of a recommender system in its tendency towards a filter bubble. The paper states that, over time, diversity was indeed diminished and thus the recommender system slightly narrowed down the items. Additionally, it was mentioned that collaborative filtering deals better with the narrowing effect than content-based filtering.

Many ideas and new developments of this thesis used the work of Christoffel [2014] as a starting point. His work engaged with an off-line performance evaluation of ranking algorithms conducted with four different datasets. Because of the great importance to measure alternative metrics to accuracy, new non-accuracy performance dimensions are introduced and evaluated. Additionally Christoffel [2014] introduced an popularity-penalizing algorithms (Section 4.4) that reached in many aspects comparable score to state-of-the-art non graph-based recommender systems. This thesis tried to extend this previous research, analyzing how diversity arises and how the problem of diversity can be addressed better.

In this section, certain important topics in recommender systems environment and related recent research were described. Because this thesis concentrates on enhancing diversity for certain users with the help of graph-based algorithms, the selected papers should help give an overview of important research. In addition, the important *cold-start* resp. long-tail problem is discussed because of its proximity and important impact on recommender systems.

3

Dataset and Metric

3.1 The Dataset

Quality and composition of the underlying dataset are important components of a recommender system. The dataset used in this thesis was published by *GroupLens*[1]. *GroupLens* collected the ratings made on the *MovieLens* website; these were made available in anonymized form for everyone. *MovieLens* has been online for more than 15 years and provides users with recommendations on new movies. For this thesis, two versions of the *MovieLens* datasets are used: a small one (100'000 ratings, referred to as *MovieLens-S*), and a larger one (1'000'000 ratings, referred as *MovieLens-B*). Two different datasets are used to provide different results and resolve the findings from one specific set. To evaluate the proposed recommender systems and new approaches, more datasets could provide more evidence and new findings. Many available open source datasets do not provide sufficient content information about the items necessary to perform some of the presented approaches, especially in Section 3.3. This was why only two, out of four, datasets Christoffel [2014] could be used.

While Nguyen et al. [2014] are able to include a more long-term evaluation of user behavior, datasets *MovieLens-S* and *MovieLens-B* do not provide ratings or associated timestamps to these ratings allowing a separation into a sufficient number of time periods. For *MovieLens-S* and *MovieLens-B*, a temporal data analysis was impossible; for the larger dataset, time span between the first rating and the last varies between 56s (user 5529) and 998 days (user 5878). The distribution of timestamps has a mean of 25 days ($\sigma = 100\ days$) and thus a temporal classification was rejected because the timestamps could not be structured to allow useful analysis. The current structure of timestamps for different users is too multi-variant to gain further insights.

The classic process to evaluate a recommender system is to separate the dataset into two subsets: a training set and a test set. The first set is used to train the recommender system and the second data subset evaluates the recommendations, calculated with the first part of the data, in relation to users' ratings. The intersection of these two sets is used as a basis to determine the quality of a recommender system for accuracy and as a basis for all metrics related to accuracy.

[1] GroupLens is a research lab at the University of Minnesota that, along with other topics, concentrates on recommender systems. *http://grouplens.org/about/what-is-grouplens/*

Before using the dataset in a recommender system, the dataset has to be analyzed and cleaned up. This preprocessing is needed because the number of ratings per user has massive influence on the outcome and evaluation of the recommender system. To minimize the effect of a different number of ratings per user, the ratings are limited to exactly **50** ratings. Before the required 50 ratings are isolated, the first 15 ratings of all users are discarded, because it is assumed that these ratings were submitted on the first visit on *MovieLens* and do not represent the usual rating behavior of a user, as Nguyen et al. [2014] stated. A new *MovieLens* user has to rate a number of **ten** movies before any recommendations are presented to him. Because these ten movies are his own choice, his tendency towards diversity is being analyzed based on the first ten movies he rated. Users with less than 65 (50 + 15) ratings cannot be considered. The ratings are sorted in temporal order and the 50 relevant ratings are included in the dataset. Following the proposition of Christoffel [2014], 35 ratings (70%) are assigned to the training dataset and the remaining 15 ratings (30%) are assigned to the test dataset.

For the *MovieLens-S* dataset, number of users is reduced by 3% to 915 (originally 943) users and number of items by 13% to 1'465 (originally 1'682) items. The *MovieLens-B* dataset was reduced to 3'717 (originally 6'040) users and 3'646 items (originally 3'900 items). This reduction of items and users is due to items not rated; these users were discarded because they submitted too few ratings.

The movie *American Beauty* was the most rated item with 1'297 ratings. The mean number of ratings per item was 35.3 ($\sigma =$ 75.8). The number of 417 movies were rated only once (13.5%) like for example the movie *Light it up.*

3.2 Defining a Metric

In this section, definition of a eligible diversity metric to find diversity-seeking users prior to the recommender system is discussed. The idea behind the definition of a diversity metric is to achieve a separation of users into a diversity-seeking and a non-diversity-seeking group.

The goal of existing metrics in the environment of recommender systems, e.g. the area under the ROC-curve (AUC-metric)[Gantner et al., 2011], is to measure the accuracy of a recommender system. The goal of these new metrics is to categorize users prior to the recommendation process in a diversity-seeking and a non-diversity-seeking group. Metrics defined in this section aid in classifying users' tendency towards diversity to adjust the recommender system in later stages and supply more diversity-seeking users with (possibly less) accurate results, but register a significant increase in diversity. Adomavicius and Kwon [2012] state that accuracy itself may not be enough; the importance of *diverse* recommendations should be emphasized. It is argued that by recommending more diverse items, more personalization is generated and user satisfaction can be increased [Adomavicius and Kwon, 2012]. Nguyen et al. [2014] research shows that the effect of narrowing recommendations differs between different groups. Narrowing in this context means less diversity in the recommendations. The goal of this new metric is to aggregate these two ideas; find what degree of diversity a user wants, in order to minimize the narrowing effect.

Since little research exists in measuring the preference for diversity prior to the recommendation process, a number of different approaches are discussed in this section. In the end, these different approaches are combined to cover different aspects of diversity. The different components of this aggregated metric are shown in the following.

In general the diversity-seeking tendency of a user is analyzed by two components: the **distance measure** and the **separation algorithm**. The different distant measures are discussed in detail in Section 3.3. In this thesis, three different measures are being used:

- Euclidean distance
- Jaccard distance
- Cosine distance

The goal is to find a distance between two items i_1, i_2. These three distance measures calculate the distance between two items in different ways.

The second component is the algorithm that analyzes the different distances of a user and his tendency towards choosing high distances over short distances. Three different algorithms to calculate this tendency of a user towards diversity are being presented, discussed in detail in Section 3.4:

- Analyzing the end of the distance distribution (**Buckets**)
- Inverse weighted buckets (**IWBuckets**)
- Diversity entropy by Shannon (**Shannon**)

To be able to assess the diversity-seeking tendency of a user for the two separation algorithms, Buckets and IWBuckets, the histogram (graphical representation of numerical data) of the pairs of distances for one single user is calculated. Generally, this *item-distance distribution* for all users is, in most cases, normally distributed (>75%, Jaccard and Euclidean Metric, Shapiro-Wilks-Test, level of significance: 0.05). In other words, users analyzed in the *MovieLens-B* dataset choose the items so that the distances between the 10 first chosen items are distributed normally and separation of users tending towards diversity cannot be made without further investigating the data. In Figure 3.1, a histogram of the distances between the first 10 chosen items of one specific user are depicted as an example of a possible *item-distance distribution*.

3.3 Distance Measures

In this section an analysis of users' diversity preferences is calculated with the help of a distance measure. Following Nguyen et al. [2014], the distances between all pairs of movies are calculated. To measure the distance, as well as the similarity between two item vectors, three different distance measures between two items are proposed in this thesis: the Euclidean distance, Cosine distance and Jaccard distance.

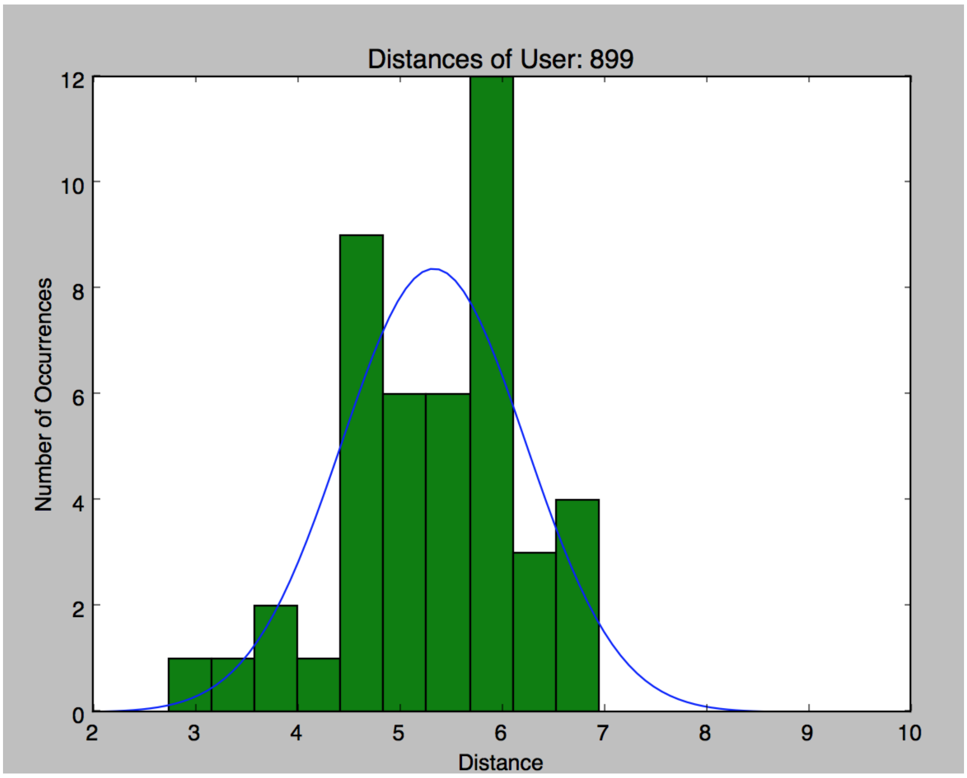


Figure 3.1: Since the first 10 items are taken into account, there exist $\frac{i*(i-1)}{2} = 45$ distances between the items. For illustration purposes the *item-distance distribution* for a random user (899) is shown. The used distance measure was the Euclidean distance.

	i_1	i_2	$\cdots$	i_{n-1}	i_n
t_1	0.1	0.2	$\cdots$	0.3	0.7
t_2	0.3	0.1	$\cdots$	0.9	0.8
$\vdots$	$\vdots$	$\vdots$	$rel(t,i)$	$\vdots$	$\vdots$
t_{n-1}	0.2	0.4	$\cdots$	0.5	0.1
t_n	0.9	0.3	$\cdots$	0.5	0.6

Table 3.1: *Tag genome* dataset visualized [Vig et al., 2012]

Because the *MovieLens* dataset with ratings contains no information about movie content, the *tag genome* dataset is used [Vig et al., 2012]. This information space delivers I a set T of tags for a set of items. In the *MovieLens* dataset, all movies are accurately described by the *tag genome* dense dataset, which consists of 1'128 weighted tags for each item. The relevance of a tag is expressed by the relevance score $rel(t,i)$, where $t \in T$ and $i \in I$, varying from 0 (not at all accurately describing the item i) to 1 (describing item i very exactly). The *tag genome* dataset can be represented as a dense matrix R, where r_{ij} matches to $rel(t,i)$. In Table 3.1, a partial matrix of tags and items is shown.

With the *tag genome* dataset, it is possible to access content information about the movies in the *MovieLens* dataset. This numerical content information allows comparison of two different items and there are different possibilities for analyzing the vector comparing the tag information of an item with another.

The *tag genome* dataset provides 1'128 tags for each movie. On one hand, some of these tags are present (relevance > 0.1) for all movies (e.g. tags *brutality*, *adaption* or *catastrophe*); on the other hand, the tag *aardman*[2] occurred only 14 times with a relevance above 0.1. The average number of occurrences is 1045 ($\sigma = 1024$) for the *MovieLens-B* dataset (6'040 movies). Because of this big deviation, a technique to classify the importance of tags is introduced. The *tf-idf* scheme presented by Gerard and Michael [1983] allows reduction of this number of tags to find the most significant tags for a specific movie. Out of all the movies, a corpus is formed with the number of occurrences of each tag (only tags with a relevance > 0.1). The frequency of this tag is compared to the tag's occurrences in the entire corpus (in a log scale and suitably normalized). The result is a *tf-idf* value for each tag in the movie. For each movie, the mean weighting is calculated and all tags above this mean are counted as relevant tags for a movie. This *tf-dif* methodology is applied to prioritize important tags for a movie and to focus on tags that define a movie more accurately. After the application of the *tf-idf* scheme, the number of relevant tags was reduced to an average of 140 tags per item ($\sigma = 44$).

[2] Aardman studios is a British animation studio

3.3.1 Euclidean Distance

One possibility to compare two item vectors is the Euclidean distance metric:

$$d_{(i_a,i_b)} = \sqrt{\sum_{c=1}^{i} [rel(t_c,i_a) - rel(t_c,i_b)]^2}$$

The Euclidean distance lends itself to use in this context instead of the cosine distance because matrix R is dense [Adomavicius and Kwon, 2012].

The mean Euclidean distance between two movies in the *MovieLens-B* dataset is 5.67 ($\sigma = 1.11$). The maximal distance between two movies in the dataset is 10.74 (*Dadetown* and *The Matrix*). *Halloween 4: The Return of Michael Myers* and *Halloween 5: The Revenge of Michael Myers* have the smallest possible difference (1.21). The largest distance between two movies effectively rated by a user in the dataset is 10.44 (*The Matrix* and *Drive Me Crazy*); one person effectively rated the *Halloween 4* and *Halloween 5*. The difference between the maximal distance and the effectively rated movie is that the maximal distance is the largest theoretically possible distance with the *MovieLens-B* dataset (6'040 movies). The effective largest distance is the distance that was effectively found as a distance for one specific user.

3.3.2 Jaccard Distance

The second distance metric used to compare two movies uses the Jaccard Metric. The Jaccard distance is given by the formula:

$$d_{(i_a,i_b)} = \frac{|T_{\forall a>\tau} \cap T_{\forall b>\tau}|}{|T_{\forall a>\tau} \cup T_{\forall b>\tau}|}$$

where $T_{\forall i>\tau}$ represents all tags t of an item i, with a value above a certain threshold τ. With the application of this formula to the set of tags, the tags with a relative high importance for both items can be set in relation to all tags of the two items.

The largest possible Jaccard distance between two movies of the *MovieLens-B* is 0.83 (*Braveheart* and *Dadetown*). The smallest Jaccard distance is between *Star Wars: Episode IV - A New Hope* and *Star Wars: Episode V - The Empire Strikes Back (1980)* with a Jaccard distance of 0.18. The largest distance between two ratings made by the same user is between *Pulp Fiction* and *The Gate of Heavenly Peace* with a Jaccard distance of 0.79. The smallest distance between two ratings made by the same user is the same as the overall minimum of Jaccard distance. The Jaccard distance has a mean of 0.51 ($\sigma = 0.037$).

3.3.3 Cosine Distance

As a third distance metric, the Cosine distance between two items is introduced:

$$d_{(i_a,i_b)} = 1 - \frac{T_{\forall a>\tau} T_{\forall b>\tau}}{||T_{\forall a>\tau}|| \, ||T_{\forall b>\tau}||}$$

where $T_{\forall i > \tau}$ represents a vector with all tags of an item i above a threshold $\tau = 0.1$.

The smallest Cosine distance between two movies was in the *MovieLens-B* dataset was 0.157 between *Wallace & Gromit: A Close Shave* and *Wallace & Gromit: The Wrong Trousers*. This combination was effectively chosen by one user. The biggest theoretical distance between two movies is 1.0 (*The Gate of Heavenly Peace* and *Adventures in Babysitting*. The effectively biggest distance a user chose was 0.99 (between *Go West* and *The Lost World: Jurassic Park*. The mean distance between two movies in the dataset is 0.80 ($\sigma = 0.11$).

3.4 Separation Algorithms

The basis of the separation in this section is either: the distance matrix, filled with one of the distances discussed above between each item (Section 3.4.1 and 3.4.2), or a completely different approach as presented in Section 3.4.3, which uses entropy diversity to separate users into a diversity-seeking and a non-diversity-seeking group.

3.4.1 Analyzing the Ends of the Distance Distribution

The basis of this separation algorithm is the *item-distance distribution*, as introduced in Section 3.2. The idea behind this separation algorithm is to analyze the histogram (*item-distance distribution*) and to identify users with particularly many high distances (it is assumed these users search more diversity than others). Numerous high distances mean the histogram is skewed to the left; more low distances mean the histogram is skewed to the right.

Because of the high number of normally distributed *item-distance distributions*, the normal distribution is used as point of origin. Distances between all of a specific user's items are distributed into equidistant buckets (e.g. ten or twenty buckets). This histogram should be normally distributed. The presented procedure analyzes the buckets with high and low distances (the *ends* of the histogram). In Figure 3.2, one sees an analysis with a total of twenty buckets, where for the high and the low distances four buckets on each side were taken into account. These buckets at the ends of the *item-distance distribution* are compared to the normal distribution and, if there are more items in the four buckets at the end, the user is classified as diversity-seeking. The normal distribution served as a baseline.

Diversity-seeking users are identified by filtering those users with a left-skewed tendency. To adjust the level, the normal distribution can be manipulated by a factor ψ. Multiplying the threshold of the normal distribution by a higher factor ψ leads to fewer diversity-seeking users in this context.

To illustrate the different parameters' behavior, number of Buckets NoB, size of the ends *ends* and ψ are varied in Table 3.2. As seen, number of buckets and number of buckets counted among the high and low distances have a far bigger effect than the parameter ψ. The number of buckets is varied between 3 and 20, because finer subdivision would not provide more useful results. The factor ψ was varied between 1.0

		$\psi = 1.0$	$\psi = 1.5$	$\psi = 2.0$
$NoB = 3$	$ends = 1$	36.4 %	35.3 %	28.5 %
$NoB = 5$	$ends = 2$	50.9 %	41.3 %	12.9 %
$NoB = 4$	$ends = 1$	24.4 %	23.8 %	23.1 %
$NoB = 8$	$ends = 3$	28.4 %	28.0 %	25.5 %
$NoB = 15$	$ends = 2$	18.9 %	17.5 %	17.2%
$NoB = 20$	$ends = 7$	37.7 %	35.5 %	26.7 %

Table 3.2: Varying parameters ψ, NoB and *ends* for the analysis procedure of analyzing the Ends of Distance Distribution.

and 2.0 empirically, keeping in mind that group of diversity-seeking users should not become too small.

3.4.2 Inverse Weighted Buckets

A second procedure to identify diversity seeking users via distances between different items uses the inverse weighted buckets (IWBuckets) method. This procedure, as the first algorithm introduced in Section 3.4.1, distributes the distances of each user into a number of buckets. Since most users have normally distributed distances, calculating the mean and variance doesn't lead to significant differences. This second approach tries to pinpoint the importance of high distances between movies, which is done by weighting the buckets inversely by their order. After the weighted distribution is calculated, the new mean value (of the weighted buckets) is calculated and set in comparison with the unweighted (adjusted due to the multiplication during the weighting process) buckets.

For this procedure, two parameters can be changed: the number of buckets NoB and a multiplication factor γ (applied on the unweighted mean). Diversity-seeking users are identified if their weighted mean is above the unweighted mean multiplied by γ. In Table 3.3, the parameters NoB and γ are varied for the *MovieLens* dataset. This table illustrates how, with a different setup of the parameters, the size of the diversity-seeking group can be manipulated depending on the goal. As seen in Table 3.3, by adjusting parameters, the size of the diversity-seeking group can be altered massively.

3.4.3 Diversity Entropy by Shannon

In all the metrics described above, the *item-distance distribution* is extremely important when analyzing a user's diversity-seeking attitude. In this section, a new separation procedure based on the entropy metric of Shannon [2001] is introduced to categorize users into a diversity-seeking and a non-diversity-seeking group. Shannon defines entropy as:

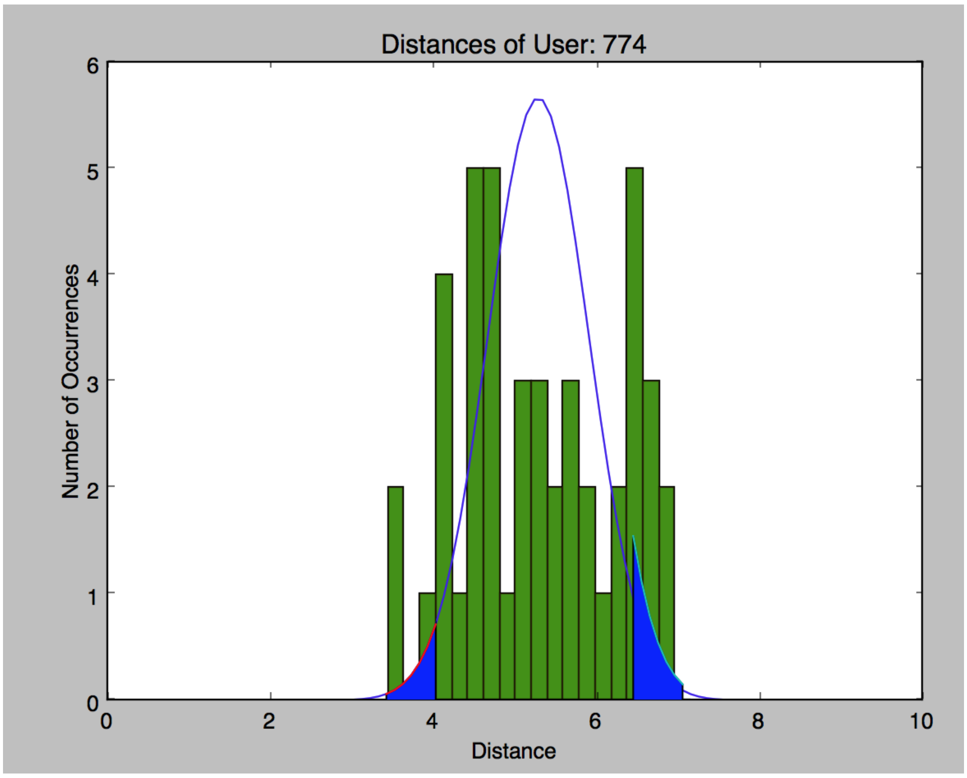


Figure 3.2: Analyzing the *item-distance distribution* (green) of user 774 in the *MovieLens* dataset. As a baseline, the normal distribution is shown along with *ends* (blue).

	$\gamma = 1.0$	$\gamma = 1.05$	$\gamma = 1.1$	$\gamma = 1.2$	$\gamma = 1.3$	$\gamma = 1.5$
$NoB = 3$	54.4 %	42.4 %	37.3 %	15.2 %	1.4 %	0.0 %
$NoB = 4$	56.7 %	46.5 %	36.4 %	17.5 %	2.8 %	0.0 %
$NoB = 5$	56.2 %	46.5 %	39.2 %	19.4 %	3.7 %	0.0 %
$NoB = 8$	57.1 %	46.6 %	40.1 %	23.5 %	6.9 %	0.0%
$NoB = 15$	57.6 %	47.0 %	39.6 %	22.1 %	9.2 %	0.0 %
$NoB = 30$	57.1 %	47.9 %	41.0 %	23.5 %	10.1 %	0.0 %

Table 3.3: For better understanding, γ and NoB are varied; the percentages correspond to the users identified as diversity-seeking. As expected, the multiplication factor γ has a huge influence on the percentage of users identified as diversity-seeking, while the effect of varying the number of buckets is negligible.

$$E_S = -K \sum_{i=1}^{n} p_i \log p_i$$

with K as a positive constant and p_i as the proportion of the ith element in correlation to the total number of members. Because Shannon used his entropy formula in the telecommunication environment, it is necessary to adapt the formula for usage for the recommender systems environment, especially in analyzing user diversity. The *tag genome* dataset can be used to modify the Shannon-entropy formula to adjust it for calculating the entropy of how diverse user's choices of movies were. As already applied in Section 3.3.2, only tags above a certain threshold τ are taken into account. This is necessary because the *tag genome* dataset is dense, but weighted. The procedure to filter out tags with a weighting below a certain threshold τ allows comparison of certain tag occurrences. The diversity entropy (Shannon) for one individual user is measured using the formula:

$$DiversityEntropy = -\sum_{i=1}^{n} \frac{occ_{t_i}}{total} \log \frac{occ_{t_i}}{total}$$

where occ_{t_i} are the occurrences of tag per user, and *total* is the number of tags a certain users has. Because not all (around 30%) of the tags are above the threshold $\tau = 0.1$ and the *total* number of occurrences of a certain tag vary significantly, the diversity entropy between each users is very different. $\tau = 0.1$ proved a feasible threshold for tag relevance. This threshold led to a mean entropy index of 31.0 (standard deviation of 4.6). The minimum diversity entropy was 16.39 and the maximum 45.81. Because this separation algorithm already measured the diversity of a user's choice, users above the mean value are counted as part of the diversity-seeking group.

3.5 Choice of Distance Metric and Separation Algorithm

Both the three distance metrics (Euclidean, Jaccard, Cosine) and the three separation algorithm offer different results with different outcomes for the recommender introduced in Chapter 4. Details about effects and results are discussed in Chapter 5.

4

Graph-based Recommender Systems

After the separation of users into diversity-seeking and non-diversity-seeking groups, this chapter introduces graph-based algorithms to be used as collaborative filtering recommender systems, focusing on different 3-path random walk algorithm alternatives. The traditional 3-path random walk operating mode is explained in Section 4.1. Section 4.2 presents the NR-RW, which includes a user's neighborhood relation to provide better results. Section 4.3 introduces a random walk algorithm based on the *tf-idf* scheme to improve results. Section 4.4 explains AN-P^3, an algorithm introduced by Christoffel [2014] that was the origin of the new algorithms.

4.1 Traditional 3-Path Random Walk

The foundations of a 3-path random walk, as, e.g., considered in Cooper et al. [2014], is an undirected bipartite graph, built from a data set, with a set of users U rating a set of items I. The union of the two entity sets users U and items I is represented by the vertices V of graph $G = (V, E)$ where $V = U \cup I$. The set of edges is represented by the relation $R \subseteq U \times I$ and for each $r \in R$, $r = \{u, i\}$ hold where $u \in U$ and $i \in I$. Because only edges exist between user u and item i, the graph is bipartite. An edge in the graph G exists if the respective entry in the user-item feedback matrix F, introduced in Section 2.1, is non-zero ($F_{ij} \neq 0$). Thus, all items rated by a user are connected with a rated item and vice versa.

To explain details of a 3-path random walk algorithm, the steps of a random walk are defined and named. The initial user u_s is the user for whom the recommendation is made; he is always the starting point for the random walk. After the first walking step, the intermediate item i_{int} is visited. The second walking step ends at the intermediate user u_{int} and the third ends at the scored item i_r. Figure 4.1 shows a traditional random walk where user 1 is u_s, item A is i_{int}, user 4 is u_{int} and item D is i_r.

The traditional 3-path random walk (**TRW**) has neither weighted edges nor weighted nodes. Algorithm 1 shows the algorithm to calculate the traditional random walk. In other algorithms the weighting w will depend on additional factors. Because TRW does not weight the scoring, $w = 1$ is being used.

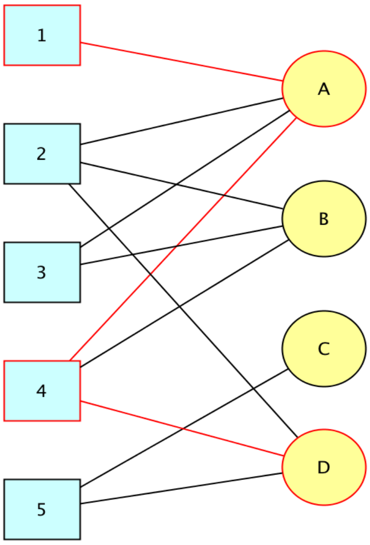


Figure 4.1: A traditional random walk without any weighting

```
Input: Graph $G$, user $u_s$, $numberOfWalks$
Result: normalized probabilities for user $u_s$ after a 3-path random walk
for all $numberOfWalks$ do
    path ← $[u_s]$;
    for 3 steps do
        currentNode ← $getRandomNeighbour(G[u_s])$;
        path.$append$(currentNode);
    end
    results[path[-1]] += $w$;
end
results ← $normalizeResults$(results);
```

Algorithm 1: Traditional 3-path random walk

4.2 Random Walk with Neighborhood Relation

In this section, a new algorithm, called neighborhood related random walk (**NR-RW**), is introduced to consider the neighborhood relationship of the two users in the random walk and try to find more diverse, but still relevant, results. The recommendation is made for the first user; the second user is the second node on the path. To measure the distance between these two different users, the Jaccard metric, already introduced in Section 3, is used again. The distance between u_s and user u_{int} is measured as follows:

$$d_{u_s,u_{int}} = \frac{|items_{u_s} \cap items_{u_{int}}|}{|items_{u_s} \cup items_{u_{int}}|}$$

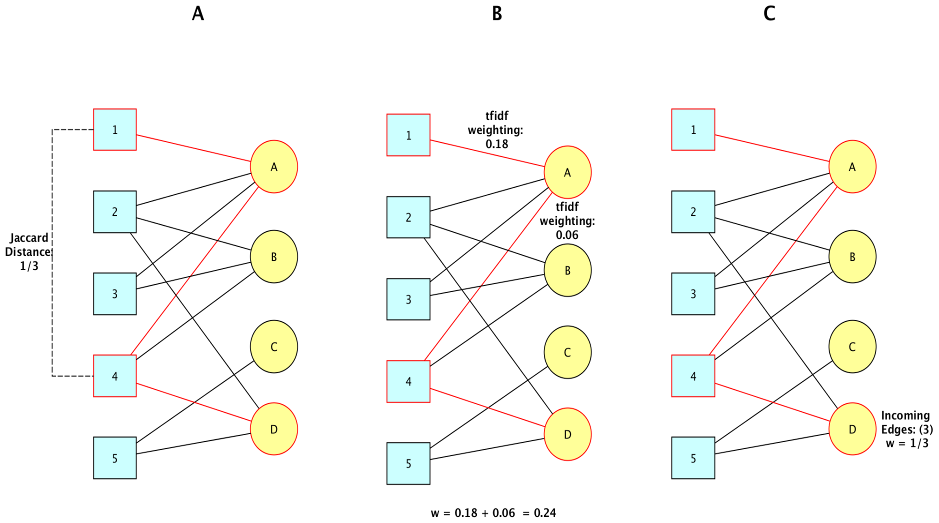


Figure 4.2: Illustrations of an exemplified NR-RW (A),TfidfRW (B) and AN-P^3 (C)

where $items_{u_i}$ are the set of items user u_i rated.

The algorithms to calculate random walks for diversity-seeking and non-diversity-seeking users are different, because each algorithm operates on its own and the end result is a ranked list of recommendations. An example of the distance calculation in a prototype graph is shown in Figure 4.2 (A). In section 4.2.1, the algorithm for non-diversity-seeking users is explained, while in 4.2.2 the algorithm for diversity-seeking users is discussed.

4.2.1 Algorithm for Non Diversity-Seeking Users

Because this algorithm aims to give diversity-seeking users more diverse results and non-diversity-seeking users less diverse results, it is applied differently to these two groups. For the non-diversity-seeking group, it is proposed that users with a small Jaccard distance are weighted higher than users with a great distance. Because similar users should have similar preferences, in theory, results should be more accurate and diversity should be diminished. The weighting factor w_{u_s,i_r} between the two users for the non-diversity-seeking group is calculated as follows:

$$w^*_{u_s,i_r} = 1 - d_{u_s,u_{int}}$$

$$w_{u_s,i_r} = \begin{cases} w^*_{u_s,i_r} & \text{if } w^*_{u_s,i_r} \geq 0.01 \\ 0.01 & \text{if } w^*_{u_s,i_r} < 0.01. \end{cases}$$

The mean distance between two users is 0.974 ($\sigma = 0.037$, $d_{max} = 1$) in the *MovieLens-B* data set. This procedure gives a high weighting for similar users and a lower rating for different users. This distinction is necessary because there are completely different users with $w^*_{u_s,i_r} = 0$; these results would not be counted. This case must be avoided, therefore a minimum of 0.01 is set.

4.2.2 Algorithm for Diversity-Seeking Users

For the diversity-seeking users, the procedure to calculate the weighting w_{u_s,i_r} is calculated differently than for non-diversity-seeking users. The goal is to weight similar and dissimilar users less than users between the two extremes. As a reference point, the mean μ of all differences between the users in the data set is calculated ($\mu = 0.974$ in *MovieLens-B*). To calculate a measurement how distant $d_{u_s,u_{int}}$ is from μ, the difference is calculated: $a_{d,\mu} = \left|d_{u_s,u_{int}} - \mu\right|$. Because $a_{d,\mu}$ can be 0, distinction of cases is also made:

$$w_{u_s,i_r} = \begin{cases} \frac{1}{a_{d,\mu}} & \text{if } a_{d,\mu} \geq 0.001 \\ 1000 & \text{if } a_{d,\mu} < 0.001. \end{cases}$$

This procedure generates the weighting for each walk which is added instead of the weighting of 1 that was for example in 2.

4.3 Alternative Algorithm with tf-idf Scheme

In this section, another algorithm, called *tf-idf* weighted random walk (**TfidfRW**), is discussed. The goal of this recommender is to improve diversity for a diversity-seeking user and, at the same time, improve accuracy for a non-diversity-seeking user. To identify popular items, the *tf-idf* scheme is applied. The idea is to analyze the intermediate item i_{int} and its importance for users u_s and u_{int}. For this algorithm, involved users' individual preference are more important than overall popularity.

The idea behind this improved algorithm is that for non-diversity-seeking users, items important for both users in the random walk are recommended with a higher probability. On the other hand, for diversity-seeking users, items less important for both involved users are recommended, with a lower probability. The idea behind this algorithm is that the individual preferences of a user are included in the *tfi-idf* weighting. Thus, a low weighting implies a low importance and a higher distance compared with strongly preferred items; these items could be interesting for a diversity-seeking user.

The formula to calculate the weighting w_{u_s,i_r} is calculated as follows, depending whether the user is diversity-seeking or not:

$$w^*_{u_s,i_r} = t_{u_s,i_{int}} + t_{u_{int},i_{int}}$$

$$w_{u_s,i_r} = \begin{cases} \frac{1}{w^*_{u_s,i_r}} & \text{if } u_s \text{ is diversity-seeking} \\ w^*_{u_s,i_r} & \text{if } u_s \text{ is not diversity-seeking.} \end{cases}$$

where $t_{u,i}$ is the *tf-idf* weight of user u and item i. w_{u_s,i_r} is applied as it is used in Algorithm 1. An exemplified iteration of the algorithm is shown in 4.2 (B) with *tf-idf* weightings of 0.18 and 0.06 which leads to $w = 0.24$.

4.4 Random Walk with Popularity Penalization

At last an algorithm introduced by Christoffel [2014] is being discussed. Christoffel [2014] discovered that the graph-based recommendation algorithms are strongly influenced by the recommended item's popularity. Because a lot of people rate popular items, the number of incoming edges is higher than for less popular items. This problem is an extension of the well-known *cold-start* problem [Schein et al., 2002]. Items not yet, or only a few times, rated are recommended less, which increases the narrowing effect and diminishes result diversity. Further, items in the long-tail would receive a higher standing and could be recommended more often. As already discussed in Section 1.1, it can be a recommender system goal to focus on less popular items (*long-tail*). To achieve this and recommend less popular items, the random walk with popularity penalization is being introduced (**AN-P^3**).

AN-P^3 performed strongly, as stated by Christoffel [2014]; its underlying idea is to penalize popular items. This extension to an ordinary random walk is called *popularity normalization. Item popularization* is a regulating factor that compensates for an item's strong popularity influence. While in TRW, the weighting of w was 1, in AN-P^3, weighting is dependent on popularity. The formula to calculate the popularity weighting of a scored item i_r is the following:

$$Q_{u_s,i_r} = P_{u_s,i_r} \times pc_{i_r}$$

where Q_{u_s,i_r} is the final normalized probability, P_{u_s,i_r} the probability calculated by the traditional 3-path random walk and pc_{i_r} is the popularity compensation factor. Christoffel [2014] proposed different ways to calculate the compensation factor pc_{i_r}. Because the evaluation in Chapter 5 showed that the effect of different ways to calculate pc_{i_r} is negligible, the factor will be calculated as follows:

$$pc_{i_r} = \frac{1}{|neighbors_{i_r}|^{\beta}}$$

where $|neighbors_{i_r}|$ is the number of neighbors item i_r has and β is the popularity penalization factor to adjust item popularity. Christoffel [2014] propose a value of $\beta =$ 0.7 as the best option to obtain an optimal trade-off between accuracy and diversity/novelty.

The algorithm for the random walk with popularity penalization looks very similar to the algorithm for the traditional one, as can be seen in Algorithm 2:

```
Input: Graph $G$, user $u_s$, $numberOfWalks$
Result: normalized probabilities for user $u_s$ after a 3-path random walk
for all $numberOfWalks$ do
    path ← [$u_s$];
    for 3 steps do
        currentNode ← $getRandomNeighbour(G[u_s])$;
        path.append(currentNode);
    end
    results[path[-1]] += w;
    cp ← $getNumberOfNeighbours$(path[-1]);
end
results ← $normalizeResults$(results)/cp;
```

Algorithm 2: 3-path random walk with Popularity Penalization

Figure 4.2 (C) shows an exemplified application of the AN-P^3 algorithm with three incoming edges for item D which leads to $w = \frac{1}{3^{0.7}} = 0.46$.

5

Evaluation

Recommender Systems are useful only if they provide suggestions and recommendations that are appreciated by the user and reflect his or her taste. Because of each user's individuality, an appropriate testing strategy must be devised to check the recommender's quality. In this section, we evaluate the introduced recommender systems, as well as the separation of users into diversity-seeking and non-diversity-seeking categories.

Section 5.1 presents the goal of this evaluation. Section 5.2 introduces some state-of-the-art recommenders used as benchmarks for the random walk-based recommenders. The recommender performance is evaluated on the basis of six metrics introduced in Section 5.3. Sections 5.5 and 5.6 analyze recommender results from different perspectives. In Section 5.7, the *Yelp* dataset is introduced as an alternative to the *MovieLens* dataset to test some results.

5.1 Goal of the Evaluation

Traditionally, recommender systems are tested by dividing the dataset into two parts: a training and a test set. The training set represents the user's previous choices and the recommender system is trained with this set. After the recommender system gives a list of recommendations for a user based on his training set, the recommended items are compared to the test dataset to test the accuracy of the recommender system.

In the *MyMediaLite* framework, two very popular evaluation metrics are implemented: the area under the ROC curve (AUC) and precision at k (Prec@k). Christoffel [2014] extended the existing framework with a variety of new metrics built to measure and evaluate the diversity found in the recommendations. The interaction of accuracy and diversity is a central research point in the recommender system environment. As Ziegler et al. [2005] stated, sacrificing accuracy for increased diversity can lead to higher user satisfaction. The goal is to answer the questions posed in Section 1.1.

5.2 Baseline Recommenders

This section measures the performance of different random-walk based recommender systems, (as introduced in Section 4), on their power to predict future user behavior.

Different metrics are applied, measuring accuracy and diversity of the predictions. To contrast these random-walk based recommender systems to other recommender systems, their performances are compared. While random walk-based recommender systems are calculated externally, reference recommender system are calculated with the *MyMediaLite* library. Recommender systems used as a reference are the weighted (**WI-knn**) k-nearest neighbor item-based collaborative filtering recommender, as well as a weighted user-based (**WU-knn**) k-nearest neighbor recommendations system. K-nearest neighbor algorithm calculates the similarity between all items in the dataset. Then, for a specific item i, the k most similar items are calculated in item i's neighborhood. The similarity in this algorithm is calculated with the Cosine distance. The predicted rating for a new item i is calculated by setting the weighting of a neighbored item in context to its rating.

Additionally, results were compared to a recommender system based on a latent factor model with matrix factorization as a basis (**BPRMF**) [Gantner et al., 2011]. BPR is the abbreviation for Bayesian personalized ranking. While matrix factorization without BPR and k-nearest neighbor algorithms are not directly optimized ranking, Rendle et al. [2009] introduces a methodology to produce a personalized ranking based on the maximum posterior estimator derived from the Bayesian analysis of the problem.

As a third group of baseline recommenders, another graph-based recommender system is introduced. Christoffel [2014] proved that, in the dimension *accuracy*, the algorithm ranking the items by the entries of the third power of the vertex transition probability matrix ($\mathbf{P^3}$) produced strong results. The basis of this $\mathbf{P^3}$ is the adjacency matrix A of graph G. The edges in G are unweighted and undirected; therefore, the entry $a_{i,j}$ is equal to $a_{j,i}$ (A is a symmetric matrix) and 1 if two edges are connected and 0 otherwise. Additionally, the degree matrix D is defined for the graph G. D is diagonal square matrix with $d_{i,i} = \sum_{j=1}^{|V|} a_{i,j}$. Multiplying D^{-1} and A produces the transition probability matrix $P = D^{-1} \times A$. The third power of the matrix P gives the probabilities of user u choosing an item i.

The results of this thesis achieved by the random walk-based algorithms are compared to structurally different recommenders, as described above. However, the comparison of the results with these recommenders must be carefully evaluated. [Christoffel et al., 2015] showed that at least 10'000 random walks must be executed for the algorithm to converge. Further, diversity metrics like Gini coefficient, Surprisal or Personalization depend especially heavily on the number of random walks. An additional effect, though not as big as the effect the number of random walks, is the effect of the number of nodes a graph possesses. Therefore, to compare the effect of the introduced algorithms, e.g. the weighted random walk or the *tf-idf* based random walk, the maximal numbers of external parameters has to be kept as constant as possible.

5.3 Evaluation Metrics

In general metrics, recommender systems pursue two different goals; on one hand, some metrics measure the *accuracy* of a recommender system, while on the other hand, metrics exist that measure the diversity in a recommendation list. To measure the *accuracy* of a recommender systems, the three most important and widely used metrics are: area under the ROC curve (AUC) and the two related metrics, precision and recall. To calculate the AUC score item, pairs of a hit (item that are covered in the test set) and no-hit are built. AUC is the number of correctly ordered item pairs in a ranked recommendation list. This number is divided by the number of all possible pairs of hit and no-hit items. The AUC is equal to the probability that a randomly drawn hit and no-hit item pair is ordered correctly relative to each other in the recommendation list[Christoffel, 2014].

To complete the metrics and produce a better picture of the recommender's accuracy, precision (**Prec@k**) and recall (**Recall@k**) are introduced. Precision (measuring exactness) and recall (measuring completeness) are defined by the formulas:

$$precision = \frac{|N_{rs}|}{|N_s|}$$

$$recall = \frac{|N_{rs}|}{|N_r|}$$

where N_{rs} are the relevant items selected, N_s the items selected and N_r the total number of relevant items available. Precision and recall are typically opposed. As precision increases, value of recall decreases. With the combination of these three accuracy-metrics, one of a recommender's primary goals can be measured. In traditional recommender system usage, only the first recommendations are taken into account; thus, $k = 20$ was chosen for use in this thesis.

All of the introduced accuracy metrics compare the ranked recommendation list and test set of a user. Therefore, scores of the metrics depend heavily on the test set. All of the three metrics, especially the AUC score, react strongly to increasing size of the test set. For the AUC score, a value around 0.83 (+/- 0.05) in the *MovieLens-S* dataset is common. To measure the quality of a recommender system, the AUC score for each user is calculated and averaged. During an analysis of the behavior of individual users, significantly different scores were found. To show the effect of the test set size, some evaluations with the *MovieLens-S* dataset are shown. An average user has a total of 108 ratings, respectively 76, in the training set and 32 in the test set. Users with a AUC score below 0.6 (17.1% of the users) have an average of 171 ratings. On the other hand, users with an AUC score above 0.93 (15.2%) have an average of 32 ratings. The number of elements in the dataset per user has a significant effect on the precision-metric (Prec@20). For example, the usual value of the precision for different recommenders is between 0.1 and 0.16. For users with a precision above 0.2 (11.4%), average number of ratings is 227; for users with a precision below 0.07 (22.3%), the average number of ratings is 54. On the basis of these numbers, one can say that the introduced scores measuring accuracy should be used as a relative metric to compare different specification of recommender systems and should not be compared to recommender results calculated with a different dataset.

The following metrics consider different aspects of diversity. A higher value of a diversity-metric indicates a higher diversity. For all metrics, overall user average is taken. A valuable metric to calculate coverage is the Gini coefficient (GiniD@20) for the top k recommendations of all users in a test set, as introduced by Adomavicius and Kwon [2012].

$$GiniD@k = 2 * \sum_{i=I} \left[\left(\frac{|I| + 1 - rank@k(i)}{|I| + 1} \right) \times \left(\frac{recCount@k(i)}{k * |U|} \right) \right]$$

where $|U|$ denotes the test set, $|I|$ the cardinality of recommendable items set and $rank@k(i)$ is the rank of item i after an ordering of the items according to their appearance in an ascending form in the top k items of all the recommendation lists. $recCount@k(i)$ denotes the number of users with an item i that appear in the top k recommendations. It is important to note that the effect of one item i on the overall coverages increases with decreasing rank of i and vice versa. This definition of the Gini coefficient differs from the original definition, which was devised to assess wealth distribution, where a lower value of the coefficient indicates a more uniform distribution. While the coefficient moves closer to 1, the cardinality of the combined recommendation lists is the same for each item i.

To measure the degree of diversity and consideration of less popular items, the surprisal (Surp@20) metric is introduced. This metric is higher for recommended items with a low popularity. With this metric, the perspective on diversity was extended to show that diversity-seeking users can be supplied with more surprising and less popular items. Surprisal is measured by the following formula:

$$Surp@k = \frac{\sum_{a=1..k} log_2 \left(\frac{|U|}{pop(i_a)} \right)}{k}$$

where U is, again, the test set of users. $pop(i_a)$ is the number of ratings that item a received during the training phase. The term $\frac{pop(i_a)}{|U|}$ measures the probability of that a user rated the item a during the training phase randomly; thus, its self-information is $log_2 \left(\frac{|U|}{pop(i_a)} \right)$. From this, the mean self-information of a user's top k items is calculated and average overall users give a value for surprisal.[Zhou et al., 2010]

The metrics to give an overview about the diversity of a recommender is completed by *Entropy-Diversity*. Adomavicius and Kwon [2012] introduce this metric as an entropy-based alternative with the formula. An modified formula of this entropy is already used in section 3.4.3:

$$DiversityEntropy = -\sum_{i=1}^{n} \frac{rec(i)}{total} \log \frac{rec(i)}{total}$$

where $rec(i)$ denotes number of users who were recommended an item i, n represents all candidate items in the recommendation list and $total$ is the aggregated number of all top k made among all users.

For the evaluation the three metrics for accuracy (AUC-score, precision and recall) and the three metrics for diversity (Gini coefficient, surprisal and diversity entropy) are being taken into account.

5.4 Evaluation Procedure

A quantitative analysis of the recommender systems conducted, e.g. Christoffel [2014], was not feasible because the goal is to compare completely different recommender systems (along with the separation of users into different groups of diversity-seeking tendency). Additionally, both accuracy and diversity metrics are compared. Therefore, a more qualitative evaluation of the results was chosen.

The procedure to evaluate the random walk-based recommendation systems – because of the various combinations of distance metric, separation algorithm and recommender system (21 possible combinations) – has to be structured clearly. The recommender system is the most important part: in a first step, recommenders are evaluated individually to demonstrate their strengths and weaknesses (Section 5.5).

In a second phase (Section 5.6), recommender performance is compared to the baseline recommenders. This section's goal is to evaluate valuable combinations of these three parts (distance metric, separation algorithm and recommender), to provide users with the best possible results. To simplify the evaluation in this phase, findings about the *MovieLens-B* dataset are presented. Should the results of *MovieLens-S* differ significantly, this would be evaluated as well and explained.

The distance measures between two items, the algorithms to separate users and the alternative random walk algorithms were inspired and built around the possibilities and features of the *MovieLens* dataset. Most parts of this evaluation analyze the performance of these datasets. In order to proof the behavior of the alternative random walk algorithms on an alternative dataset, in Section 5.7 the *Yelp* dataset is introduced and evaluated. Because of this dataset's structure, a distance between two individual items cannot be measured. Therefore user separation is not conducted, but the overall performance of the dataset can be evaluated.

5.5 Random walk based Algorithms

In this Section the performance of NR-RW (Section 5.5.1), TfidfRW (Section 5.5.2) and AN-P^3 (Section 5.5.3) are presented individually.

5.5.1 Neighborhood Related Random Walk

Table 5.1 shows the results for the metrics defined in Section 5.3 performed with NR-RW. It must be remembered that the algorithms behave differently for diversity-seeking and for non-diversity-seeking users (Section 4.2).

The results clearly show that the separation into two groups had a massive effect on the accuracy metrics (AUC, precision and recall). For all combinations of distance measure and separation algorithm, these metrics are higher for the non-diversity-seeking group. The best performance was achieved with the combination Euclidean distance and IWBuckets with an AUC score of 0.883 and a precision of 0.09. The worst performance concerning accuracy for the non-diversity-seeking group was the separation by Shannon

	Distance	**SepAlgo**	**#users/#items**	**AUC**	**Precision**	**Recall**	**Entropy**	**Gini**	**Surprisal**
DS user	Cosine (Tfidf)	Buckets	1987 / 2595	0.814	0.06	0.08	4.33	0.02	3.02
	Cosine (Tfidf)	IWBuckets	2298 / 2673	**0.820**	0.06	0.08	4.31	0.02	3.02
	Euclidean	Buckets	1476 / 2347	0.779	0.05	0.07	**4.67**	**0.04**	3.00
	Euclidean	IWBuckets	1449 / 2341	0.777	0.05	0.07	4.63	0.04	2.98
	Jaccard (Tfidf)	Buckets	1896 / 2544	0.809	0.06	0.08	4.42	0.03	**3.03**
	Jaccard (Tfidf)	IWBuckets	2208 / 2619	0.816	0.06	0.08	4.36	0.02	3.01
		Shannon	2360 / 2385	**0.820**	**0.07**	**0.09**	4.10	0.02	2.59
Non DS user	Cosine (Tfidf)	Buckets	1730 / 2451	0.876	0.09	0.12	5.76	0.11	3.78
	Cosine (Tfidf)	IWBuckets	1419 / 2318	0.872	0.09	**0.13**	5.69	0.11	3.69
	Euclidean	Buckets	2241 / 2684	0.881	0.08	0.11	5.91	0.12	4.71
	Euclidean	IWBuckets	2268 / 2667	**0.883**	0.09	0.11	5.89	0.11	4.66
	Jaccard (Tfidf)	Buckets	1821 / 2526	0.875	0.09	0.12	5.72	0.11	4.78
	Jaccard (Tfidf)	IWBuckets	1509 / 2405	0.870	0.09	0.12	5.66	0.11	4.99
		Shannon	1357 / 2546	0.865	0.08	0.11	**6.29**	**0.18**	**5.90**

Table 5.1: Accuracy and diversity performance of the NR-RW algorithm. *Distance* represents the distance metric that was applied. *SepAlgo* the user separation algorithms where *IWBuckets* is the Inverse Weighted Buckets algorithm introduced in Section 3.4.2. For the metrics *Precision*, *Recall*, *Entropy* and *Gini* $k = 20$ was chosen.The best results for each metric are bold.

Index (AUC: 0.865, precision: 0.08 and recall: 0.11). For accuracy, separation with the Shannon algorithm performed less well than the rest, but this separation algorithm performs well in diversity. For all three distance measures, the diversity metrics (Entropy, Gini and surprisal) are significantly higher for the separation by the Shannon algorithm than by the algorithm analyzing buckets.

One unexpected factor was the NR-RW algorithm for the non-diversity-seeking users that performed better in diversity than the algorithm for diversity-seeking users. The algorithm was designed to weight random walks via a user, with a certain distance from the starting user u_s higher to build a more diverse recommendation list. Because of the bad performance of NR-RW with diversity-seeking users, NR-RW (in the version for non-diversity-seeking users) was applied to the dataset of diversity-seeking users. This result (Distance: Cosine (Tfidf), Separation Algorithm: Shannon, AUC: 0.871, precision: 0.09, recall: 0.12, Entropy: 5.23, Gini: 0.07, Surprisal: 3.21) shows that the separation of the dataset itself – not just the different algorithms – has an influence.

Overall it can be said that the separation of users together with NR-RW did not lead to the desired results, but improved both accuracy as diversity for the non diversity-seeking users.

5.5.2 Tfidf Ranked Random Walk

Using the TfidfRW algorithm provides a new perspective on the dataset. Table 5.2 shows the results for this recommender system. In contrast to NR-RW, this recommender system performs similarly in terms of accuracy both for the diversity-seeking group as well as for the non diversity-seeking group despite a completely different algorithm.

For a better understanding of the TfidfRW algorithm, the evaluation was repeated,

	Distance	**SepAlgo**	**#users/#items**	**AUC**	**Precision**	**Recall**	**Entropy**	**Gini**	**Surprisal**
DS user	Cosine (Tfidf)	Buckets	1987 / 2595	0.860	0.08	0.11	4.91	0.04	3.29
	Cosine (Tfidf)	IWBuckets	2298 / 2673	**0.864**	0.08	0.11	4.93	0.04	**3.30**
	Euclidean	Buckets	1476 / 2347	0.850	0.09	0.12	4.70	0.04	3.01
	Euclidean	IWBuckets	1449 / 2341	0.847	0.09	0.12	4.65	0.04	2.99
	Jaccard (Tfidf)	Buckets	1896 / 2544	0.859	0.08	0.11	4.94	0.05	3.28
	Jaccard (Tfidf)	IWBuckets	2208 / 2619	**0.864**	0.08	0.11	**4.95**	**0.05**	**3.30**
		Shannon	2360 / 2385	0.857	0.09	0.12	4.46	0.03	2.75
Non DS user	Cosine (Tfidf)	Buckets	1730 / 2451	0.866	0.08	0.11	5.25	0.07	3.40
	Cosine (Tfidf)	IWBuckets	1419 / 2318	0.861	0.09	0.12	5.18	0.07	3.32
	Euclidean	Buckets	2241 / 2684	0.873	0.08	0.11	5.37	0.07	4.31
	Euclidean	IWBuckets	2268 / 2667	**0.875**	0.08	0.11	5.37	0.07	4.29
	Jaccard (Tfidf)	Buckets	1821 / 2526	0.865	0.08	0.11	5.16	0.06	4.39
	Jaccard (Tfidf)	IWBuckets	1509 / 2405	0.859	0.08	0.11	5.12	0.06	4.61
		Shannon	1357 / 2546	0.856	0.08	0.11	**5.85**	**0.12**	**5.55**

Table 5.2: Accuracy and diversity performance of the tfidfRW algorithm. *Distance* represents the distance metric that was applied. *SepAlgo* the user separation algorithms. For the metrics *Precision*, *Recall*, *Entropy* and *Gini* $k = 20$ was chosen. The best results for each metric are bold.

using the opposite version that the algorithm offered, in Section 4.3.

5.5.3 Random Walk with Popularity Penalization

AN-P^3 as introduced in Section 4.4 is similar to TRW but penalizes popular items. The popularity is defined by the number of incoming edges of the scored item. Christoffel [2014] stated that the algorithm provides the best results for $\beta = 0.7$.

In Table 5.3, results for AN-P^3 are shown. Overall, results are promising because, despite a slightly lower accuracy, the diversity metrics could be substantially increased. Especially with the Gini coefficient, the algorithm achieved extremely high results (Gini: 0.6).

As demonstrated, performance of the diversity-seeking group is, for all six metrics, slightly weaker than for the non-diversity-seeking group. As already seen in Section 5.5.1, separation led to a higher performance in both accuracy and diversity for the non-diversity-seeking group. The best accuracy performance was achieved using the separation using the Shannon Index.

For AN-P^3, the same algorithm was used for both datasets; separation into two groups does have a significant effect. The group with diversity-seeking users performed less strongly than the complementary group. Also, for this algorithm, it is remarkable that the diversity-seeking group didn't perform better for the diversity metrics than the non-diversity-seeking group.

5.5.4 Summary of Individual Analysis

Individual analysis of the three random walk-based algorithms was conducted for a more thorough analysis of individual strengths and weaknesses. Better recommendations for

	Distance	SepAlgo	#users/#items	AUC	Precision	Recall	Entropy	Gini	Surprisal
DS user	Cosine (Tfidf)	Buckets	1987 / 2595	0.816	0.03	0.04	7.41	0.53	8.45
	Cosine (Tfidf)	IWBuckets	2298 / 2673	0.818	0.03	0.04	**7.45**	0.53	**8.55**
	Euclidean	Buckets	1476 / 2347	0.808	0.03	0.05	7.29	0.54	8.22
	Euclidean	IWBuckets	1449 / 2341	0.802	0.03	0.05	7.29	0.53	8.27
	Jaccard (Tfidf)	Buckets	1896 / 2544	0.818	0.03	0.05	7.39	0.53	8.37
	Jaccard (Tfidf)	IWBuckets	2208 / 2619	0.820	**0.04**	0.05	7.44	0.53	8.40
		Shannon	2360 / 2385	**0.833**	**0.04**	0.05	7.44	**0.60**	8.18
Non DS user	Cosine (Tfidf)	Buckets	1730 / 2451	0.827	0.04	0.06	7.38	0.55	9.38
	Cosine (Tfidf)	IWBuckets	1419 / 2318	0.825	0.05	0.07	7.33	0.57	9.48
	Euclidean	Buckets	2241 / 2684	0.828	0.04	0.05	**7.49**	0.54	9.26
	Euclidean	IWBuckets	2268 / 2667	0.831	0.04	0.05	7.47	0.54	9.20
	Jaccard (Tfidf)	Buckets	1821 / 2526	0.822	0.04	0.05	7.40	0.54	9.61
	Jaccard (Tfidf)	IWBuckets	1509 / 2405	0.822	0.04	0.06	7.34	0.55	**9.68**
		Shannon	1357 / 2546	**0.836**	**0.07**	**0.09**	7.48	**0.60**	8.35

Table 5.3: Accuracy and diversity performance of the AN-P^3 algorithm with parameter $\beta =$ 0.7. *Distance* represents the distance metric that was applied and *SepAlgo* the user separation algorithms. For the metrics *Precision*, *Recall*, *Entropy* and *Gini* $k = 20$ was chosen. The best results for each metric are bold.

users can only be generated with a proper understanding of distance metric, separation algorithm and random walk algorithm behavior.

Importantly, choice of the recommender is crucial and has the strongest effect on the performance. Also observed was the fact that, for all three recommenders, diversity for the non-diversity-seeking group was higher than for the diversity-seeking group, across all recommenders. This is to that effect interesting, considering that the separation of the users is carried out with different distance metrics and different separation algorithms. Nevertheless, the algorithms produce more diverse results for the non-diversity-seeking users, without reducing accuracy.

From the individual analysis results, it was clear that Euclidean distance tends to increase the accuracy for non-diversity-seeking users (e.g. NR-RW and AN-P^3), while at the same time, performs badly for diversity-seeking users. As expected, the two separation algorithms, based on distributing the users to buckets, perform much in the same way across all three recommenders.

To simplify the evaluation process in Section 5.6 for the three recommenders, two combinations of distance and separation algorithm each (one diversity-seeking and one non-diversity-seeking) are chosen. The combinations are shown in Table 5.4. For diversity-seeking users (DS), chosen criteria were high diversity with a reasonable accuracy (all three metrics combined). For non-diversity-seeking users (NDS), the criteria were high AUC score, as well as high precision.

In addition to the new algorithms presented (NR-RW, TfidfRW, AN-P^3), for each baseline recommender the best combination of distance metric and separation algorithm are added in Table 5.4 to simplify the evaluation across algorithms.

Overall, performance after the separation of diversity-seeking and non-diversity-seeking users was not as expected. For all algorithms (both newly introduced and baseline

		Recommender	Distance	SepAlgo	AUC	Prec@20	Recall@20	Entropy	Gini
MovieLens-B	*DS*	NR-RW	Euclidean	Buckets	0.779	0.05	0.07	4.67	0.04
		TfidfRW	Cosine (Tfidf)	IWBuckets	0.864	0.08	0.11	4.93	0.04
		AN-P^3		Shannon	0.833	0.04	0.05	7.44	0.60
		TRW	Euclidean	Buckets	0.854	0.09	0.12	4.88	0.05
		P^3	Jaccard (Tfidf)	IWBuckets	0.860	0.09	0.12	4.78	0.04
		BPRMF	Cosine (Tfidf)	IWBuckets	0.854	0.07	0.10	5.99	0.12
		WI-knn	Jaccard (Tfidf)	Buckets	0.835	0.08	0.11	6.40	0.20
		WU-knn	Jaccard (Tfidf)	IWBuckets	0.838	0.08	0.11	5.84	0.11
	NDS	NR-RW	Euclidean	IWBuckets	0.882	0.09	0.11	5.89	0.11
		TfidfRW	Euclidean	IWBuckets	0.881	0.08	0.11	5.37	0.07
		AN-P^3		Shannon	0.836	0.07	0.09	7.48	0.60
		TRW	Euclidean	IWBuckets	0.873	0.08	0.11	5.18	0.06
		P^3	Euclidean	IWBuckets	0.883	0.08	0.11	5.13	0.05
		BPRMF	Euclidean	IWBuckets	0.862	0.07	0.10	5.95	0.10
		WI-knn	Euclidean	IWBuckets	0.850	0.08	0.11	6.44	0.19
		WU-knn	Cosine (Tfidf)	IWBuckets	0.852	0.10	0.13	5.53	0.09

Table 5.4: Chosen combination for all algorithms to simplify evaluation. Best options for diversity-seeking users (DS) are above the line and best options for non-diversity-seeking users are below the line (NDS).

recommenders), the AUC score, precision and recall were reduced. Following Ziegler et al. [2005], sacrificing accuracy for higher diversity in recommendations will improve user satisfaction. While acting on the assumption that a trade-off exists between accuracy and diversity for most recommender systems, data acquired in this thesis does not fully support this claim. The not random walk-based recommenders (BPRMF, WI-knn and WU-knn) follow the stated assumption. After the separation, accuracy metrics for diversity-seeking users are lower than for the non-diversity-seeking users. At the same time, metrics for diversity are higher for diversity-seeking than non-diversity-seeking users.

For the random walk-based algorithm, the situation is different. Across these recommenders, the performance – both in accuracy and diversity – is better for the non-diversity-seeking users.

5.6 Overall Performance Evaluation

This section compares best options for each algorithm with the baseline recommenders. In Section 5.6.1, comparison is done without a separation, according to diversity-seeking tendency, while Section 5.6.2 and Section 5.6.3 evaluate the behavior of separated groups.

5.6.1 Recommenders without Diversity Separation

In this section, recommenders without a separation in diversity-seeking and non-diversity-seeking users are evaluated; these results, with the two datasets, are presented in Table 5.5. For the algorithms with different applications for diversity-seeking and non diversity-seeking users (e.g. NR-RW and TfidfRW), the non-diversity-seeking version was chosen because this algorithm performs more accurately, as evaluated in Section 5.5.

	Recommender	AUC	Prec@20	Recall@20	Entropy@20	Gini@20	Surprisal
MovieLens-B	NR-RW	**0.892**	0.09	0.12	5.80	0.10	3.85
	TfidfRW	0.883	0.08	0.11	5.27	0.06	3.46
	AN-P^3	0.855	0.05	0.07	**7.65**	**0.61**	**7.65**
	TRW	0.882	0.08	0.11	5.06	0.05	3.34
	P^3	**0.893**	0.09	0.11	5.01	0.04	3.32
	WU-knn	0.849	0.09	0.12	5.92	0.10	3.97
	WI-knn	0.865	0.08	0.11	6.38	0.16	4.57
	BPRMF	0.871	0.07	0.10	5.99	0.11	4.04

Table 5.5: Alternative algorithms along with baseline recommenders without a separation of users according to their diversity-seeking tendency. Above the dotted line are the new algorithms introduced in Section 4, below the line the baseline recommenders.

As observed, without user separation into different groups, NR-RW performs notably better than the traditional random walk for both datasets. For the smaller dataset *MovieLens-S*, NR-RW improves accuracy (AUC and Prec@20), at the same time increasing diversity metrics like Entropy@20, Gini@20 and Surprisal. NR-RW, in comparison to the other baseline recommenders WI-knn and BPRMF, performs less well, both in accuracy and diversity. TfidfRW produces similar results to the traditional random walk (TRW), but slightly better in diversity. P^3 performance is similar to TRW and TfidfRW.

The superior performance of WI-knn, WU-knn and BPRMF vs. the random walk based algorithms like Christoffel [2014] could not be reproduced with this dataset. NR-RW and P^3 accuracy performance were strong. The baseline recommenders couldn't compete in accuracy, but perform more strongly in diversity metrics. Of the random walk-based algorithms, NR-RW performed powerfully; TfidfRW seemed to be a moderate improvement compared with the TRW. The small gain of TfidfRW for the diversity metrics compared to TRW could be explained because less popular items are recommended more often. Very popular items had a lower *tf-idf* weighting and are thus recommended less than in TRW. At the same time, TfidfRW's accuracy was identical to TRW.

AN-P^3 algorithm performance was different from all other algorithms. AN-P^3 had, in both datasets, significantly lower accuracy scores (Prec@20 = 0.05 and Recall@20 = 0.07), but diversity performance was better (Entropy = 7.65 and Gini@20 = 0.61); the Gini coefficient, especially, was remarkably high.

5.6.2 Diversity-Seeking Users

As it can be seen in Table 5.4, Jaccard (Tfidf) was the distance metric that produced the best results for diversity-seeking users. The IWBuckets (Inverse Weighted Buckets) separation algorithm delivered the best separation for performance of the diversity-metrics. The best algorithm for defined diversity metrics is AN-P^3 (Entropy = 7.44 and Gini =

0.60); this performance could be explained by Ziegler et al. [2005].

The other introduced algorithms, (NR-RW, TfidfRW) were weaker in diversity than AN-P^3, or the baseline recommenders. For diversity-seeking users, the baseline recommenders, especially BPRMF, seemed to be a good choice. The algorithms performed strongly in both accuracy and diversity, with a minimal trade-off.

5.6.3 Non Diversity-Seeking users

Table 5.4 shows the performances of all algorithms for non-diversity-seeking users. Most of the best algorithm performances for the algorithms were produced with the Euclidean distance metric and IWBuckets used as a separation algorithm. Separation with the Shannon algorithm performed well only for AN-P^3.

For non-diversity-seeking users, random walk based algorithms performed better than the baseline recommenders. NR-RW, in particular, showed a strong accuracy performance (AUC = 0.882, Prec@20 = 0.09) and, at the same time, strong diversity scores (Entropy@20 = 5.80, Gini@20 = 0.11). P^3 was similar in accuracy performance to NR-RW, but NR-RW outperformed P^3 with more diversity in the recommendations. It seemed that forming a neighborhood relation between two users was more valuable if one desired increased accuracy rather than relying on *popularity normalization*, as applied in AN-P^3. On the other hand, TfidfRW (AUC = 0.881, Prec@20 = 0.08, Entropy@20 = 5.37, Gini@20 = 0.07) performed accurately with a lower diversity than NR-RW, but higher than traditional random walk (Entropy@20 = 5.18, Gini@20 = 0.06).

5.7 Evaluation of Alternative Dataset

Both user separation with different distance measures and extension of the graph-based random walk algorithms were conducted based on the findings achieved by *MovieLens* dataset analysis. In this section, results collected in Section 5.5 and Section 5.6 are compared to an alternative dataset. The chosen dataset is from *Yelp*[1] and includes about 1'569'264 reviews from 366'717 users about 60'786 businesses. It is important to note that the dataset contains no information comparable to the *tag genome* dataset available for *MovieLens*. Because of the large number of users compared to the businesses, the mean user reviewed only 4 businesses ($\mu = 4.3$, $\sigma = 15.1$); one could not apply a collaborative filtering strategy to define the distance between two items. Thus, user separation was abandoned.

To use the dataset in the same way as *MovieLens*, the minimum number ratings per user was set to 20. The reviews in the *Yelp* dataset could be about many diverse businesses, e.g. doctor's office, stores, or gyms. Because a recommendation makes more sense for a business category, the dataset in this thesis was reduced to only businesses only characterized as a restaurant. Setting a minimum of reviews from a user and taking only restaurants into account, the dataset was reduced to 6'335 users, 18'585 items and 224'926 reviews.

[1] http://www.yelp.com/dataset_challenge

	Recommender	**AUC**	**Prec@20**	**Recall@20**	**Entropy@20**	**Gini@20**	**Surprisal**
Yelp	NR-RW	0.888	0.04	0.08	6.96	0.09	6.28
	TfidfRW	0.887	0.04	0.07	6.79	0.08	6.22
	AN-P^3	0.875	0.01	0.02	**8.84**	**0.39**	**11.26**
	TRW	0.885	0.04	0.08	6.22	0.04	5.87
	P^3	**0.919**	0.04	0.08	5.96	0.03	5.76
	WU-knn	0.841	0.04	0.08	6.40	0.04	5.96
	BPRMF	0.873	0.03	0.05	6.17	0.04	5.94

Table 5.6: Alternative algorithms along with baseline recommenders for the *Yelp* dataset with users above 20 reviews and businesses that are restaurants. The dataset consists of 6'335 users and 18'585 items. Above the dotted line are the new algorithms introduced in Section 4, below the line the state-of-the-art recommenders. WI-knn was not feasible because of the great number of items. The best results for each metric are bold.

The results of the evaluation of the *Yelp* dataset are shown in Table 5.6. For the *Yelp* dataset, graph-based random walk algorithms showed a strong performance. The highest AUC-score was indeed achieved by P^3 recommender, but NR-RW, TfidfRW and TRW had the highest score of the remaining recommenders. NR-RW again proved to be a valuable alternative to improve accuracy and diversity. TfidfRW is slightly weaker both in accuracy and diversity, but proves itself to be better in both departments than the traditional random walk. The AN-P^3 performed well in the diversity department and also achieved high accuracy (Entropy@20 = 8.84, Gini@20 = 0.39 and Surprisal = 11.26).

The analysis of this second dataset showed that the performance of the alternative random walk algorithms introduced in Section 4 do indeed have the potential to improve existing algorithms; AN-P^3 and NR-RW are excellent alternatives.

6

Conclusions

Recommender systems have become an important component in structuring Internet data and finding relevant information. Information or items are recommended in a trade-off between accuracy and diversity. It is not sufficient to just recommend similar items; it has become essential that user satisfaction is maximized by recommending items with surprisal, diversity and by serendipity. Research shows, as explained in Section 2.2, that the quality of a recommender is as well defined by its potential to provide the user with diverse results as with accurate recommendations.

This work split the recommendation process in two main phases: a separation and characterization of users into a more diversity-seeking and a less diversity-seeking group and the recommendation process itself, adjusted to the individual preference of the user. To evaluate the diversity-seeking tendency of a user, the first 10 choices of movies are analyzed, all choices made before any recommendations are presented to the user. During this phase content information of the items are included to enrich the recommendation and provide better results. Three different measures to calculate the distance between two individual items are presented along with three separation algorithms to classify a user as diversity-seeking or not. In a second phase three recommendation systems are presented with the goal to provide better results for both groups.

The results clearly showed that the separation algorithm had a smaller effect on results than the distance measure choice. After choosing the best options for all recommender systems and the separation of groups (Table 5.4), 75% of best results were achieved by the algorithm of inverse weighted buckets. It is remarkable that for the non-diversity-seeking users, only for the AN-P^3 (which performed poorly for this group of users), another separation algorithm was chosen. Overall, it seems that IWBuckets is a valuable choice to separate users.

The three distances measures (Euclidean, Cosine and Jaccard) showed different strengths and weaknesses during the evaluation. While Jaccard and Cosine (both had the *tf-idf* scheme applied on tags) performed especially well for the diversity-seeking group, Euclidean provided the better results for non-diversity-seeking users. It is assumed that the Euclidean measure of distance, in combination with IWBuckets, provides good results because the effective distance between users is better reflected than by using Cosine or Jaccard distance. Interestingly, the other two measures perform better for diversity-seeking users. One can deduce that these distance measures better reflect the diversity-seeking tendency of users than the Euclidean distance.

The evaluation of these different elements is conducted by six quality metrics for recommender systems (AUC score, precision and recall measuring the accuracy while Gini, Diversity Entropy and surprisal measure the diversity/surprisal). Christoffel [2014] combined many metrics in his work, analyzing both accuracy and diversity. From this variety of quality metrics for recommender systems, the most important were chosen to provide a simpler overview and reduce redundant information, a by-product of applying too many metrics. The evaluation was conducted with two different versions of the widely used *MovieLens* dataset. The evaluated task was an item-ranking task on two datasets, with implicit feedback enriched by content information about the items. The results were compared by the metrics with numerous well-developed baseline recommenders (k-nearest neighbor, BPRMF, P^3).

Without user separation into diversity-seeking and non-diversity-seeking groups, performance of the baseline recommenders was strong overall. The nearest neighbor algorithms and BPRMF showed especially high scores for the diversity and surprisal metrics, while maintaining high accuracy. Traditional random walk, as well as P^3 (based on a very similar approach: TRW uses simulation, while P^3 calculates by increasing transition probability to the third power) perform more accurately, but less well in diversity/surprisal. Because of P^3's high computation costs, as stated by Christoffel [2014], using this may not be the optimal solution. Without separation, two of the algorithms introduced in Section 4 performed strongly. On one hand, in terms of accuracy, NR-RW is as strong as P^3, but without the matrix calculation necessary. Most interestingly, NR-RW contradicts the theory that there is a trade-off between accuracy and diversity. NR-RW is better, or equal, in almost all metrics, compared to TRW or BPRMF. On the other hand, AN-P^3 shows significantly better diversity and surprisal results than any other recommender tested. AN-P^3 is slightly less accurate than the other recommender systems. Additionally, classical recommender systems (k-nearest neighbor, BPRMF) performed strongly overall.

Some conclusions drawn from the *MovieLens* dataset can be confirmed by analyzing the *Yelp* dataset. It could be affirmed that the performance of P^3 was strong in accuracy, but had high computational costs. Aside from P^3, the best performance was achieved by NR-RW for accuracy and AN-P^3 for diversity. These two algorithms offer a real improvement compared to the existing recommender systems, due to the poor performance of the not graph-based for this dataset.

Separation of users into diversity-seeking and non-diversity-seeking groups had an strong influence on recommendations. Throughout all recommender systems and with different separation algorithms, as well as different distance measures, the separation led to significantly better results for the non-diversity-seeking group for the accuracy metrics and the diversity metric, an unexpected result. Christoffel [2014] identified a general trade-off between accuracy and diversity, a finding that could not be supported by applying the graph-based recommenders to the separated dataset. On the other hand, the non-graph-based baseline recommenders performed as expected and had an increase in accuracy for the non-diversity-seeking group while simultaneously decreasing in diversity/surprisal metrics. It is therefore assumed that the separation algorithm works better for non-graph-based recommenders. The accuracy metrics results for the combined dataset (diversity-seeking and non-diversity-seeking users) were, for all applied recommenders, better than after the separation. For the diversity metrics, the non-diversity-seeking users achieved the best results on average.

It is particularly interesting that the metrics for the non-separated algorithms were, on average, better than after the separation. There are multiple reasons why this could be so; one explanation is, of course, the larger number of users and items. When using the *MovieLens-S* dataset, results were even worse; graph size obviously has an influence on performance.

In Section 5.1, two central questions are posed on how combinations of distance measures, separation algorithms and recommender systems influence recommendations. The first research question – whether a separation of the users into different levels of diversity has an impact on performance metrics – can be answered positively. The results across all test setups and with all different combinations of separation algorithms and distance measures can be confirmed. It is also clear that a separation favors the group of non-diversity-seekers because that is where results could be most improved. It seems that the group of diversity-seeking users is not as easy to satisfy.

The second research question revolves around what useful combination of components would improve user satisfaction. The answer could be that, for the diversity-seeking users, the AN-P^3 algorithm should be used. This algorithm provides significantly better results for diversity metrics than any other for creating diverse and surprising results. For the non-diversity-seeking group, NR-RW performed more accurately than the rest, despite performing well for the diversity metric. It can be argued that non-diversity-seeking users do not want diversity, but current research shows that users do indeed want certain diversity, but the non-diversity-seeking users value accuracy more than diversity.

7

Limitations and Future Work

After the conclusion in Chapter 6, this chapter discusses limitations of, and difficulties for, designing and evaluating a recommender system (Section 7.1), as well as some ideas about future work and research (Section 7.2).

7.1 Limitations

This thesis analyzes different graph-based recommender systems based on groups of users separated by their tendency towards diversity. The topic of recommender systems is complex; many different parameters can substantially influence quality of outcome. This section explains some of the elements that make recommender system analysis difficult and limit this work.

One of the biggest limitations was having only one suitable dataset, in different versions, to evaluate all aspects of this thesis available. Because of the goal to combine content-based information with collaborative filtering, it was essential to find a dataset that had both a user-item feedback matrix and some information about item content. Despite the fact that various datasets exist to evaluate recommender systems, e.g. Flixster[1], there is no additional content-based information available. Therefore the *Yelp* dataset was only used to evaluate the modified recommender systems without a separation of users.

During the evaluation of recommender systems, it became clear that comparing different research, with completely different dataset compositions, is very difficult. These compositions have a huge influence on the performance of the metrics, in both accuracy and diversity. During the experiments, it became obvious that a user with 400 ratings in the training dataset and 170 ratings in the test dataset performs much worse than a user with just 20 ratings in the training dataset and 7 ratings in the test dataset. Because it was, in some cases, difficult to figure out the composition and details of an evaluation dataset for other research, a simple comparison between metrics scores could not be conducted. To minimize the effects of external factors like size of the datasets, the number of ratings was kept constant for all users.

The metrics introduced to calculate the quality of recommender systems (both accuracy metrics and diversity metrics) measured very similar things. These metrics are

[1] http://socialcomputing.asu.edu/datasets/Flixster

very useful and, in theory, measure different aspects of the results but the reality of the evaluation showed a high correlation of the metrics for their specific purpose (accuracy and diversity). A lower correlation would have allowed to draw more conclusions about individual recommenders and separation algorithms.

7.2 Future Work

This, and related work, present opportunities for research in many interesting directions. Some ideas are presented in this section.

Separation of users concentrated on the content information gained by analyzing the *tag genome* dataset (Section 3.3). This dataset represents only one possibility to characterize items. In future work, analysis could focus on whether gathering information from other sources, e.g. by using Wikipedia[2] entries to characterize items provides a better way to calculate the distance between two items and the separation of users.

The separation algorithm itself is a first step to the goal of providing the optimal recommender system for a user. In this work, a user is assigned either to the diversity-seeking or a non-diversity-seeking group. One way to improve this (rather strict) distribution is to find a measure of users' diversity-seeking tendency. Parallel, recommender systems are sorted according to their ability to produce diversity or accuracy. According to the diversity-seeking preference, a recommender system that best fits individual user preferences may produce the most user satisfaction.

The evaluation in Chapter 5 showed that recommenders exist that perform better than baseline recommenders, in some respects, e.g. NR-RW or AN-P[3]. Therefore it may be useful to develop these newly introduced algorithms further to improve results. For NR-RW, a simple technique to calculate the distance between two users was applied (Jaccard distance). It may be possible to include state-of-the-art networking analysis, for example that developed by Yang and Leskovec [2013], to formulate a better understanding of a user's social network and improve recommendations.

Another opportunity may lie in further elaboration of the term 'diversity', in the context of recommender systems. This work assumes, based on previous research by, for example, Adomavicius and Kwon [2012], or Herlocker et al. [2004], that diversity is important to measure the quality of a recommender system. Pariser [2011] states that 35% of Amazon sales are generated by recommender systems. This illustrates the immense value of these systems and why further enhancements are absolutely necessary. The term 'diversity' is not yet fully defined and further work on this topic is essential. Definition is needed about what kind of diversity is useful for a recommender system. Because topics like this are so important for Amazon, it is necessary for them to produce good results without too much experimenting. On the other hand, a streaming service like Spotify can suggest very diverse items without much risk, because a user can just skip over a bad recommendation.

[2] https://www.wikipedia.org/

References

Adomavicius, G. and Kwon, Y. (2012). Improving aggregate recommendation diversity using ranking-based techniques. *Knowledge and Data Engineering, IEEE Transactions on*, 24(5):896–911.

Brynjolfsson, E., Hu, Y. J., and Smith, M. D. (2006). From niches to riches: Anatomy of the long tail. *Sloan Management Review*, 47(4):67–71.

Christoffel, F. (2014). Recommending Long-Tail Items with Short Random Walks over the User-Item-Feedback Graph. Master's thesis, University of Zurich.

Christoffel, F., Paudel, B., Newell, C., and Bernstein, A. (2015). Blockbusters and wallflowers: Accurate, diverse, and scalable recommendations with random walks. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 163–170. ACM.

Cooper, C., Lee, S. H., Radzik, T., and Siantos, Y. (2014). Random walks in recommender systems: exact computation and simulations. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 811–816. International World Wide Web Conferences Steering Committee.

Desrosiers, C. and Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer.

Fleder, D. and Hosanagar, K. (2009). Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Management science*, 55(5):697–712.

Fouss, F., Faulkner, S., Kolp, M., Pirotte, A., Saerens, M., et al. (2005). Web recommendation system based on a markov-chainmodel. In *ICEIS (4)*, pages 56–63.

Gantner, Z., Rendle, S., Freudenthaler, C., and Schmidt-Thieme, L. (2011). Mymedialite: A free recommender system library. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 305–308. ACM.

Gerard, S. and Michael, J. M. (1983). Introduction to modern information retrieval.

Gori, M., Pucci, A., Roma, V., and Siena, I. (2007). Itemrank: A random-walk based scoring algorithm for recommender engines. In *IJCAI*, volume 7, pages 2766–2771.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53.

Lee, S., Park, S., Kahng, M., and Lee, S.-g. (2012). Pathrank: a novel node ranking measure on a heterogeneous graph for recommender systems. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1637–1641. ACM.

McNee, S. M., Riedl, J., and Konstan, J. A. (2006). Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI'06 extended abstracts on Human factors in computing systems*, pages 1097–1101. ACM.

Nguyen, T. T., Hui, P.-M., Harper, F. M., Terveen, L., and Konstan, J. A. (2014). Exploring the filter bubble: the effect of using recommender systems on content diversity. In *Proceedings of the 23rd international conference on World wide web*, pages 677–686. International World Wide Web Conferences Steering Committee.

Pariser, E. (2011). *The filter bubble: What the Internet is hiding from you.* Penguin UK.

Park, Y.-J. and Tuzhilin, A. (2008). The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 11–18. ACM.

Rajaraman, A. and Ullman, J. D. (2011). *Mining of massive datasets.* Cambridge University Press.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009). Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press.

Ricci, F., Rokach, L., and Shapira, B. (2011). *Introduction to recommender systems handbook.* Springer.

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

Schein, A. I., Popescul, A., Ungar, L. H., and Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260. ACM.

Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55.

Vig, J., Sen, S., and Riedl, J. (2012). The tag genome: Encoding community knowledge to support novel interaction. *ACM Transactions on Interactive Intelligent Systems (TIIS)*, 2(3):13.

Yang, J. and Leskovec, J. (2013). Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 587–596. ACM.

Zhang, Y. C., Séaghdha, D. Ó., Quercia, D., and Jambor, T. (2012). Auralist: introducing serendipity into music recommendation. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 13–22. ACM.

Zhou, T., Kuscsik, Z., Liu, J.-G., Medo, M., Wakeling, J. R., and Zhang, Y.-C. (2010). Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences*, 107(10):4511–4515.

Ziegler, C.-N., McNee, S. M., Konstan, J. A., and Lausen, G. (2005). Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, pages 22–32. ACM.

List of Figures

List of Tables