University of Brighton

Hydra: A hybrid framework that improves collaborative filtering recommendation quality

Gharbi Khamis Alshammari

A thesis submitted in partial fulfilment of the requirement of the University of Brighton for the degree of Doctor of Philosophy

December 2018

Abstract

Recommender systems are a viable solution against information overload found on daily used online services, rapid message exchanges and extended variety and complexity in data volumes. Recommender systems are widely employed to help individuals to swiftly iterate through multiple available options to find products or services that are likely to be of interest to them, e.g. news items, scholar articles or entertainment options and can be used to provide suggestions based on past user interactions, similar content selections, demographic data using artificial intelligence or machine learning methods.

There are various methods for providing recommendations and the two most widely used are collaborative and content-based filtering that rely on the similarity of items. Collaborative filtering draws upon the ratings that a user has previously given whereas content-based recommendations are formed according to knowledge from relevant context.

Hybrid recommendation methods or frameworks can use two or more recommendation algorithms to improve the quality of the provided recommendation. However, hybrid recommender systems in domains with high information overload, fuzziness and uncertainty are rare. This thesis proposes a novel framework to improve recommendations including the long-tail recommendation problem by applying a case-based reasoning approach based on user history. This method is extended with a multi-level algorithm that works as an add-on to existing collaborative filtering algorithms. A novel similarity measure for item-based collaborative filtering has been developed by integrating the triangle similarity measure with a multi-level method, considering the length and the angle

of rating vectors among users. Finally, a feature combination method is applied which combines user-based collaborative filtering with attributes of demographic filtering.

The proposed framework has been thoroughly evaluated using well known datasets like MovieLens and Yahoo! Movies and metrics such as mean absolute error (MAE) and root mean squared error (RMSE). The results validate the framework by showing that prediction accuracy is improved, while outperforming all the algorithms used as a baseline.

Acknowledgments

Approaching the end of the journey of my PhD, I would like to take an opportunity to thank all the people who helped me reach this goal. I would like to express my appreciation to my parents, my wife, my kids and all my family for their support, encouragement, trust, love and motivation throughout my study.

A special thank goes to my great supervisors, Prof. Miltos Petridis, Dr Stelios Kapetanakis and Dr Nikolaos Polatidis, for their expert guidance and helpful assistance, and for being very kind advisors and mentors for my PhD study.

My sincere thanks also go to my friend Jose L. Jorro-Aragoneses and his supervisors, Prof. Juan A. Recio-Gracia and Prof. Belen Diaz-Agudo from Universidad Complutense de Madrid, for their generous support and comments on my research.

Last but not least, I would like to thank the University of Brighton for supporting me in conducting this study and providing me with the facilities to create this thesis.

Declaration

I declare that the research contained in this thesis, unless otherwise formally indicated within the text, is the original work of the author. The thesis has not been previously submitted to this or any other university for a degree, and does not incorporate any material already submitted for a degree.

Abstract ii

Acknowledgments iv

Declaration v

List of Figures xi

List of Tables xiv

List of Equations xiv

Abbreviations xvi

1 Introduction 1

1.1 Motivation 3

1.2 Research questions 4

1.3 Aims 5

1.4 Objectives 6

1.5 Main contribution to knowledge 6

1.6 Thesis outline 8

1.7 List of publications 9

2 Literature review 11

2.1 Introduction 11

2.2 Recommender systems 12

2.2.1 Formalisation of the recommender systems problem 14

2.2.2 Motivation 15

2.2.3 The aim of recommender systems 16

2.2.4 The importance of recommender systems 16

2.2.5 Major challenges of recommender systems 17

2.2.5.1 Sparsity 17

2.2.5.2 Scalability 18

2.2.5.3 Diversity 18
2.2.5.4 Cold start problem 19
2.3 User profile 19
2.3.1 User information 21
2.3.2 Item information 23
2.4 Personalisation 24
2.5 Recommendation methods 27
2.5.1 Collaborative filtering 27
2.5.1.1 User-based CF 30
2.5.1.2 Item-based CF 30
2.5.1.3 Advantages of CF 31
2.5.1.4 Major challenges in CF 31
2.5.2 Content-based filtering 33
2.5.2.1 Related work 34
2.5.2.2 Advantages of CBF 36
2.5.2.3 Major challenges in content-based filtering 36
2.6 Demographic filtering 37
2.7 Hybrid method 38
Weighted 39
Switching 39
Mixed 40
Feature combination 40
Cascade 40
Feature augmentation 41
Meta-level 41
2.8 Case-based reasoning 42
2.8.1 The CBR Principles 43
2.8.2 Related work 44

2.9 Multi-level method 45

2.10 Multi-criteria 47

2.11 Prediction algorithms 48

2.12 The long tail recommendation problem 51

2.12.1 Definition 52

2.12.2 Related work 54

2.12.3 Difficulties of long tail recommendation 55

2.13 Recommender system evaluation 56

2.13.1 Offline evaluation 57

2.13.2 Online evaluation 58

2.13.3 User study 60

2.14 Summary 61

3 Research Methodology 63

3.1 Introduction 63

3.2 Switching hybrid multi-level recommender system framework 63

3.2.1 CF module based on users 66

3.2.2 Content-based module based on user history 67

3.3 Rating prediction 68

3.4 Recommendation algorithms 69

3.4.1 k-Nearest Neighbour (k-NN) 69

3.5 Similarity measures 72

3.5.1 Cosine similarity 73

3.5.2 Pearson correlation coefficient (PCC) 73

3.5.3 Euclidean distance 74

3.6 Baseline user-based CF 74

3.7 Summary 74

4 Experimental evaluation 76

4.1 A switching CBR experiment 76

4.1.1 MovieLens 100K Dataset 77

4.1.2 Experiment setup 81

4.1.3 Collaborative filtering ..... **Error! Bookmark not defined.**

4.1.4 Content-based filtering ..... **Error! Bookmark not defined.**

4.1.5 Combined method 68

4.1.6 CBR method ..... **Error! Bookmark not defined.**

4.2 Switching multi-level method 82

4.2.1 Multi-level 83

4.3 Triangle similarity using the multi-level method for item-based CF 83

4.4 Real dataset 86

4.4.1 MovieLens 100K 86

4.4.2 MovieLens 1M 86

4.4.3 Yahoo! Movies 87

4.5 Offline validation 87

4.6 Demographic attributes using k-NN, Random Forest, neural Network and AdaBoost 88

4.6.1 The k-Nearest Neighbour (k-NN) 90

4.6.2 Random Forest 91

4.6.3 Neural Network 91

4.6.4 AdaBoost 91

4.7 Cross-validation 91

4.7.1 Leave-one-out 92

4.7.2 k-fold 92

4.8 Evaluation metrics 92

4.9 Summary 94

5 Results and discussion 95

5.1 Introduction 95

5.2 Switching CBR experiment results 96

5.2.1 Collaborative filtering 97

5.2.2 Content-based 97

5.2.3 Combined method 97

5.2.4 CBR approach 97

5.2.5 Experimental results 97

5.3 Switching multi-level experiment results 103

5.3.1 Experimental results 105

5.3.2 Discussion 112

5.4 Triangle similarity with multi-level algorithm results. 113

5.4.1 PCC 114

5.4.2 Multi-level CF 115

5.4.3 Experimental results 115

5.4.4 Discussion 121

5.5 A demographic feature combination using k-NN and Random Forest.. 122

5.5.1 Experimental results 122

5.5.2 Discussion 127

5.6 Summary 128

6 Conclusions and future work 130

6.1 Conclusions 130

6.2 Future work 132

List of Figures

Figure 1: User-based collaborative filtering 28

Figure 2 Item based collaborative filtering 31

Figure 3: Content-based filtering 34

Figure 4: CBR cycle (Aamodt & Plaza, 1994) 43

Figure 5: The long tail of rating distribution.....53

Figure 6: A switching hybrid multi-level recommender system framework 64

Figure 7 MAE results with different value of δ.....98

Figure 8: User x movies sparsity.....78

Figure 9: Distributions of ratings by user and movie.....79

Figure 10 Histogram distribution of item's rating frequency using 100.000 MovieLens dataset 80

Figure 11: A triangle in three-dimensional (3D) space (S. Sun et al., 2017).....85

Figure 12: Feature combination architecture 89

Figure 13: Accuracy rate based on Euclidean distance similarity 98

Figure 14: Accuracy rate based on Pearson similarity.....99

Figure 15: MAE base on Euclidean distance similarity.....100

Figure 16: MAE based on Pearson similarity 100

Figure 17: RMSE based on Euclidean distance similarity.....101

Figure 18: RMSE based on Pearson similarity 101

Figure 19: Improvement rate for the CBR system based on Euclidean distance similarity 102

Figure 20: Improvement rate for the CBR system based on Pearson similarity.....102

Figure 21 the observed similarity results using Polatidis equation.....104

Figure 22 the observed similarity results using the propsed equation 104

Figure 23: MAE results for MovieLens dataset using 70/30 test.....105
Figure 24: RMSE results for MovieLens dataset using 70/30 test.....106
Figure 25: MAE results for MovieLens dataset using 60/40 test.....106
Figure 26: RMSE results for MovieLens dataset using 60/40 test.....107
Figure 27: The improvement rate over all methods 107
Figure 28 MAE results for the MovieLens 1.000.000 dataset 108
Figure 29 RMSE results for the MovieLens 1.000.000 dataset 109
Figure 30 MAE results for the MovieLens 1.000.000K dataset 109
Figure 31 RMSE results for the MovieLens 1.000.000 dataset 110
Figure 32 The MAE improvement rate for movieLens 1.000.000 dataset over all methods.....111
Figure 33 The RMSE improvement rate for movieLens 1.000.000 dataset over all methods.....111
Figure 34: MAE results for the MovieLens 100K dataset 116
Figure 35: RMSE results for the MovieLens 100k dataset 116
Figure 36: MAE results for the MovieLens 1M dataset 117
Figure 37: RMSE results for the MovieLens 1M dataset 117
Figure 38: MAE results for Yahoo! Movies dataset.....118
Figure 39: RMSE results for Yahoo! Movies dataset.....119
Figure 40 The improvement rate for movieLens 100.000 dataset over all methods 120
Figure 41 The improvement rate for movieLens 1.000.000 dataset over all methods 120
Figure 42 The improvement rate for Yahoo movie! dataset over all methods 121
Figure 43: MAE results for the MovieLens dataset using k-NN 123

Figure 44: MAE results for the MovieLens dataset using Random Forest.....124

Figure 45: MAE results for the MovieLens dataset using NN and AdaBoost.....124

Figure 46: RMSE results for the MovieLens dataset using k-NN 125

Figure 47: RMSE results for the MovieLens dataset using Random Forest.....125

Figure 48: RMSE results for the MovieLens dataset using NN and AdaBoost.....126

Figure 49: MAE improvement rate 126

Figure 50: RMSE improvement rate 127

List of Tables

Table 1: Information type and method of data collection. 21

Table 2: Benchmark prediction algorithms using 100k MovieLens dataset.....48

Table 3: Benchmark prediction algorithms using 1M MovieLens dataset 49

Table 4: MovieLens dataset information 78

List of Equations

(1)…… 47

(2)…… 65

(3) …… 65

(4)… 66

(5) …… 66

(6)…… .....……66

(7)…… 66

(8)……...67

(9)……...68

(10)……68

(11)……68

(12)……73

(13)……74

(14)……74

(15)……74

(16)……89

(17).. 82

(18)……85

(19). 85

(20)…….94

(21)…….94

(22)……...102

Abbreviations

CBF	Content-Based Filtering
CBR	Case-Based Reasoning
CF	Collaborative Filtering
CNN	Convolutional Neural Network
CTR	Clik-Through Rates
DF	Demographic Filtering
DRL	Deep Reinforcement Learning
HS	Hybrid System
IBCF	Item Based Collaborative Filtering
IDF	Inverse Document Frequency
k-NN	k-Nearest Neighbour
LTRP	Long Tail Recommendation Problem
MAE	Mean Absolute Error
MF	Matrix Factorisation
MLP	Multi Layer Perception
NN	Neural Network
PCC	Pearson's Correlation Coefficient
RF	Random Forest
RMSE	Root Mean Squared Error
SVD	Singular Value Decomppsition
TF	Term Frequency
UBCF	User Based Collaborative Filtering

1 Introduction

There has recently been explosive growth in the availability of information on the Internet. As a result of this growth, users face information overload problems, with the availability of a wide variety of content that may potentially serve their interests. There are a lot of options to be considered to resolve the problem of finding the best way for a particular user to access appropriate information. The growth of available information and products leads to complexity in information filtering and it is often a challenge in terms of time to find the right information. Recommender systems are a key step in tackling this issue. Users have trouble handling large volumes of information, and with cognitive and data sparsity, when attempting to find appropriate information at the right time (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013). Recommender systems are the most popular approach to filtering information and helping users find items that are unseen by the user, thus maximising the provision of successful user suggestions. These suggestions usually rely heavily on profiling, user demographic information and user behaviour analysis by applying data mining and machine learning techniques.

At present, two recommendation strategies are most prominent: collaborative filtering (CF) and content-based filtering (CBF). An example of CBF may be an online image gallery where users can provide suggestions either explicitly or implicitly. A user profile can be generated and gradually populated with the preferences of users and, as a consequence, becomes steadily able to provide more accurate recommendations. Such an approach relies not only on knowledge acquisition but also on ongoing maintenance by the designers of the system. This is

due to the fact that the knowledge of the recommender system does not exist in the first place. CF, on the other hand, is dependent on the observation that item knowledge is not required because the other user profiles may be able to 'suggest' a number of 'right' items to a user whose profile is sufficiently similar to any profile available within a pool of users. The degree of similarity can be calculated based on users ratings from past events, items and opinions, which are related to the target users.

Several recommender systems focus on the highest ratings of popular or long-standing items. This works well when there are sufficient numbers of users to rate each of the items. However, there are some cases in recommender systems such as items in the long tail that need to be considered in order to produce more accurate recommendations and alleviate sparsity in the data. Furthermore, a challenge emerges when an item has not been rated 'enough', i.e. when there are insufficient user ratings available. This is known as the long tail problem in recommender systems LTRP, and typically refers to less popular or newly added items. Such items belong to the distribution long tail (Park, 2013). They should not, therefore, be ignored because they could solve the cold start problem, where item ratings are sparse. Park (2013) presented a solution to the LTRP by splitting the recommendation into a head and tail and providing a clustering approach for tail items. Park and Tuzhilin (2008) use an approach in which similar users are clustered with similar clusters in order to mitigate the CF data sparsity challenge. Park (2013) also employs clustering so as to improve the accuracy of predictions of CF; items are divided into similar groups and CF is applied to each group separately.

The only method in the literature that utilises algorithm switching was proposed by Ghazanfar and Prugel-Bennett (2010). Their method switches between item-based collaborative filtering and the Naïve Bayes collaborative filtering classifier. The authors use the Naïve Bayes classifier as a second phase to be implemented when item-based filtering fails to predict a rating.

This thesis presents a novel approach to the LTRP by relying on past user experience to retrieve items with similar ratings for a new user. This is done by means of a switching method between collaborative filtering and content-based filtering in order to mitigate the increased error rate of a recommender system. A user history is used as a case base, which is identified using case-based reasoning (CBR) in conjunction with a hybrid CF-CBF recommender system. In addition, Multi-level method is used with the switching method that considers the number of co-rated items to indicate the degree of connection between users. Furthermore, the triangle similarity is utilised with the multi-level method to enhance the similarity of the users and improve the accuracy of the recommendations.

The proposed model and experiments on a freely available dataset of more than 100,000 cases have yielded promising results and, in a number of cases, compare favourably to the CBR and CF baseline, as well as to the approaches presented in the empirical literature.

1.1 Motivation

In recommender systems there are several complex issues that have not yet been fully resolved. One of the most important is producing an accurate recommendation for users. Another is formalising quality user/item relationships from sparse data.

It is often the case that a CBR system is combined with other classifier/prediction algorithms to improve the accuracy and performance of the system. For instance, a CBR system was integrated with support vector machine algorithms to predict business failure by Li and Sun (2009). Hence, with the advantages of CBR cycles and features, an effective method is proposed of benefiting from these advantages of the CBR system to solve the LTRP and improve recommendation accuracy. In addition, a multi-level method that improves prediction accuracy through modifying the similarity measures was integrated with the switching method to further improve accuracy in long tail recommendation. Furthermore, The advantage of triangle similarity raises the important issue of modifying multi-level similarity using the triangle instead of the Pearson. It considers both the length and the angle between two vectors.

1.2 Research questions

The research questions of this thesis are:

1. What is a suitable solution to the long tail problem as encountered in collaborative filtering?

2. Can recommender systems increase their accuracy by using hybrid approaches? This question can be broken down to the following sub-questions:

    2.1 To what extent can an algorithm that switches between collaborative filtering and content-based filtering improve predictive accuracy?

    2.2 How can demographic information and item descriptions be used to enhance user profiles and improve predictive accuracy?

2.3 To what extent do triangle similarity measures improve predictive accuracy using a multi-level method?

2.4 How can a multi-level method be used to improve switching methods in terms of predictive accuracy?

1.3 Aims

The aim of this research was to develop a switching hybrid method that is able to improve prediction accuracy compared to current recommendation algorithms. In addition, this thesis addresses some of the challenges that recommender systems face when applying a single technique and to address these issues by applying two or more techniques. With a hybrid system, both user profile and content information are obtained as knowledge bases. This knowledge is utilised as a case base to find and retrieve an optimal solution, thereby making a significant contribution to the long tail problem in recommender systems. Furthermore, benefits will be realised for both user profile representation and item description, leading to better relations between users and items and helping to achieve more accurate recommendations. A multi-level method was integrated with the switching method, leading to significant improvement in the recommendations compared with the traditional baseline method. In addition, a new method is presented that improves item-based CF using a multi-level method with triangle similarity. Demographic filtering attributes are then combined with user-based collaborative filtering, that stated the importance of demographic attributes in prediction, utilising the k-Nearest Neighbour and Random Forest classifiers.

1.4 Objectives

Recommender systems have been a widely debated research topic since the early 1990s. Research in this area has been even more active since the inauguration of the Netflix prize, which was about improving the predictive accuracy of the baseline recommendation algorithm by 10%. Several methods were proposed to solve this problem:

1. Identifying user 'failing' in prediction with each method individually. This includes user-based CF and CBF through utilising a deep analysis and observing the different in each method.

2. Address the weaknesses of each individual method by modelling a novel switching method and devising an efficient way of using the strength of both approaches, which improves over-all results.

3. Utilise demographic attributes and user-based CF to support user profile representation to address CF limitations in regards to finding similar users.

4. Apply the multi-level method using triangle similarity measures and compare this similarity with PCC meaure using different datasets.

5. Improve the discoverability of long tail items and help users expand their horizons.

1.5 Main contribution to knowledge

Addressing the challenges of current recommender systems, this research makes a contribution to knowledge by improving the accuracy of recommendations and solves some of the limitations that exist with CF and CBF. To the best of my

knowledge, this research is the first switching hybrid recommender system that applies case-based reasoning and multi-level algorithms to address the long tail recommendation problem. The use of the switching method has many advantages. Firstly, this method does not require any pre-calculation of items, as do a number of methods such as clustering, that use pre-calculation steps to categorise items. Another contribution is that there is no need in this system to create any other information. If the collaborative filtering algorithm cannot calculate the rating based on the users, it uses the user's profile history, which is the same information that is utilised in content-based filtering. In addition, the proposed algorithms increase accuracy when predicting the items that belong to the long tail, whereas many other methods see a decrease in accuracy when recommending long tail items. Furthermore, this research addresses the importance of changing the similarity measures to improve prediction in recommender systems. These contribution can be outlined as follows:

1. This research provides an effective method of solving the problem of items in the long tail (Alshammari et al., 2017).

2. This thesis examines how knowledge (a combination of user profile and item information) can improve recommendation accuracy using CBR, which makes a significant contribution to the LTRP (Alshammari et al., 2017).

3. It presents a new approach that uses triangle similarity with a multi-level method. It enhances the similarity using pre-defined levels that meet the common co-rating items, which further improves the accuracy of recommendations (Alshammari, Kapetanakis, Polatidis, & Petridis, 2018).

4. It examines the multi-level method using a switching technique and compares it with the existing methods. The switching method adopts a new similarity measure, whereas all the existing studies that have been explored in the literature focus on traditional similarity measures such as Pearson correlation and cosine similarity (Alshammari, Jorro-aragoneses, Kapetanakis, Polatidis, & Petridis, 2019).

5. This thesis makes a contribution to the study of demographic filtering along with the collaborative filtering approach by applying a feature combination hybrid method and comparing four different classifiers (Alshammari, Kapetanakis, Polatidis, Petridis, & Alshammari, 2018).

1.6 Thesis outline

This section outlines the structure of the thesis, through an overview of each task. The chapters proceed as follows:

- Chapter 1 provides a general introduction to the problem, stating the research questions, aims, objectives and the main contribution to knowledge.

- Chapter 2 provides a systematic review of related work. A general overview of state-of-the-art recommender systems is first presented. The chapter then details the recommendation methods used in existing works that are utilised in the thesis. Finally, this chapter discusses how results have been examined and evaluated in the literature.

- Chapter 3 presents the research methodology and the system followed in the design of the experiments. This chapter details all the recommendation

algorithms that were used as a baseline, such as traditional CF and CBF and similarity measures.

- Chapter 4 presents the evaluation methods and measures used to validate the contribution of the thesis.

- Chapter 5 provides the experiment results and discusses the contribution of the proposed method of a recommender system that addresses the long tail recommendation problem.

- Chapter six presents the conclusion and discusses possible future work.

1.7 List of publications

The relevant publications that have been produced based on this thesis are as follows:

Conferences:

- Alshammari, G., Jorro-Aragoneses, J. L., Kapetanakis, S., Petridis, M., Recio-García, J. A., & Díaz-Agudo, B. (2017, June). A Hybrid CBR Approach for the Long Tail Problem in Recommender Systems. In *International Conference on Case-Based Reasoning* (pp. 35–45). Springer, Cham.

- Alshammari, G., Kapetanakis, S., Alshammari, A., Polatidis, N., & Petridis, M. (2018, September). A Hybrid Feature Combination Method that Improves Recommendations. In *International Conference on Computational Collective Intelligence* (pp. 209–218). Springer, Cham.

- Alshammari, G., Kapetanakis, S., Polatidis, N., & Petridis, M. (2018, September). A Triangle Multi-Level Item-Based Collaborative Filtering Method that Improves

Recommendations. In *International Conference on Engineering Applications of Neural Networks* (pp. 145–157). Springer, Cham.

- Alshammari, G., Jorro-Aragoneses, J. L, Polatidis, N., Kapetanakis, S., & Petridis, M. (2019) A switching approach that improves prediction accuracy for long tail recommendations.Intelligent System conference, 2019. (Accepted).

Journals:

- Alshammari, G., Jorro-Aragoneses, J. L, Polatidis, N., Kapetanakis, S., & Petridis, M. A switching multi-level method for the long tail recommendation problem. Journal of Intelligent and Fuzzy Systems, 2019. (Accpted).
- Alshammari, G., Kapetanakis, S., Alshammari, A., Polatidis, N., & Petridis, M. Improved movie recommendations based on a hybrid feature combination method. Vietnam Journal of Computer Science, 2019 (Accpted).

2 Literature review

This chapter presents the background theory that is related to recommender systems. It covers subjects related to recommender system techniques and their advantages and limitations. It begins with a general definition of recommender systems, their aims, and the importance of studying and researching this domain. It then goes into more detail about each method that has been utilised to solve the long tail recommendation problem (LTRP), such as collaborative filtering, content-based filtering, case base reasoning, demographic filtering and hybrid systems. The LTRP is explored in more depth via the relevant literature, and the importance of this problem in recommender systems and the difficulties faced by existing methods are explored. The multi-level method is clarified via a recent study that employs this method in recommender systems. Finally, the evaluation method is also explained in more detail in this chapter.

2.1 Introduction

Recommender systems are an essential tool for users since they can help find the most relevant items and suggest the most interesting items that are specific to the user. Nowadays, most recommender system applications can be found in entertainment domains such as music, video and movies (Herlocker, Konstan, Terveen & Riedl, 2004a). There has been substantial work in this field, largely focussing on the two most popular recommender systems: collaborative filtering (CF; Linden, Smith & York, 2003) and content-based filtering (CBF; Semeraro, 2009). In CF, recommender systems rely on the ‘wisdom of crowds’ and are based on the assumption that similar users should have the same preferences, and as a result give the same rating to similar items. CBF, on the other hand, is based on

item descriptions in relation to users, as well as on their preferences as they relate to information retrieval and filtering techniques, e.g. TF-IDF, which presents a weight score of terms that occur within the document and (vector space) representation (Bhadoria, Sain, & Moriwal, 2011). For example, (Rousseau, Browne, Malone, & Ó Foghlú, 2004) applied TF-IDF to build a personalised user profile in E-learning domain.

This chapter introduces the theoretical background to recommender systems, outlines the definitions relevant to the main contribution of this research and reviews relevant previous research. The challenges faced by recommender systems are outlined in detail. The chapter then provides the background to recommender systems. The main recommender system methods are then explained in more detail and compared in terms of accuracy and performance, in order to justify the use of the algorithm used in this work, and to illustrate each method and show how different they are from each other, to provide a benchmark for the results obtained from the proposed method. Finally, similarity measures are outlined.

2.2 Recommender systems

Recommender systems can be defined as adaptable tools that can help users search for, filter and classify information, and help users find relevant items (Resnick & Varian, 1997). They can assist users by suggesting products that are similar to those they are looking at. They involve techniques that provide suggestions for items that are most likely to be of interest to a particular user (Adomavicius, Rokach & Shapira, 2015). Such systems are software agents that elicit the interests and preferences of individual consumers and make recommendations accordingly. They have the potential to support and improve the quality of the decisions consumers

make while searching for and selecting products online (Se, Haracteristics, Mpact, Xiao & Benbasat, 2007).

Recommender systems were first researched in the mid-1990s, relying on the idea that users share similar items or opinions, thereby helping to make recommendations to others (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994). Researchers established a collaborative filtering technique based on a ratings structure. The most common formulation was to calculate ratings for items that had not yet been seen by a particular user. (Malone, 1986) provide an idea of how artificial intelligence helps people to share relevant information, whereas in (Resnick & Varian, 1997) the authors pointed out that in reality people often ask others to recommend something based on their experience. (Burke, 2002) described EntreeC as a system that personalises recommendations based on the user's interests. The system can thus effectively guide users to an interesting product. To develop such a recommender system, developers have implemented many systems with different domains to improve the recommendations. However, there remain some limitations that need to be improved in order to enhance the quality of the recommendations provided.

Recommender systems try to predict what the most suitable products or services are likely to be, based on the user's preferences and constraints. In order to complete such a computational task, recommender systems collect information from users regarding their preferences, which are either explicitly expressed (e.g. as ratings for items) or inferred by interpreting the actions of the user. For instance, recommender system may consider navigation to a particular product page as an implicit sign of preference for the items shown on that page (Adomavicius et al., 2015).

2.2.1 Formalisation of the recommender systems problem

A recommender system has two main attributes that are used as input to the system: user and item. Users give their ratings/likes to a specific item/product. Users are indicated by $U = \{u_1, u_2, \ldots\ldots\ldots, u_n\}$, where n is the number of users using the recommendation system. Items are indicated by $I = \{i_1, i_2, \ldots\ldots\ldots\ldots\ldots, i_m\}$, where m represents the number of items being rated. Usually, each user has a profile that contains a list of items. This list represents the items the user has rated in the past. The item also has some content; for example, in regards to movies, this could be genre, director, actors and so on.

Finally, the items that have been rated by two users u and v, for example, $I_u$and I_v, are an important concept in the presented method, and $I_{u,v}$ has been used to denote this concept. In a similar fashion, $U_{i,j}$ is used to denote the set of users that have rated both items i and j.

Two of the most important problems associated with recommender systems are the prediction problem, which is about the estimation of item ratings for a given user, and the top-N recommendation problem, which is associated with recommendation rankings. The first problem consists of finding, for a particular user u, the new item $i \in I$ in which u is most likely to be interested. When ratings are available, this task is most often defined as a regression (or multi-class) classification problem, where the goal is to learn a function that predicts the rating of user u for a new item i. This function is then used to recommend to the active user u_a an item i for which the estimated rating has the highest value.

Recommender systems can save and update the history of each user based on past interaction and behaviour within the system. Then, unknown items that match the target user are recommended. However, in order to generate personalised recommendations, users are required to rate some items in the beginning. As in, for example, the MovieLens web recommender system, a new user has to rate some items to get relevant recommendations from the system.

2.2.2 Motivation

Knowledge about users and items can help a system to recommend interesting items by reasoning the user's requirements. Thus recommender systems can tackle the information overload problem by filtering and matching aspects that are likely to be of interest to particular users. More specifically, user profiles are applied to predict an item based on similar users with similar knowledge that have rated this item in the past. In addition, such systems apply the search history of users to identify their interests and preferences.

Recommender systems work very well at predicting the most suitable items that are relevant to the user's interests. They have been applied in many different domains such as e-learning, e-tourism, e-government and e-commerce (Lu, Wu, Mao, Wang & Zhang, 2015).

Recommender systems are able to personalise recommendations by considering similarity functions. On-demand applications can benefit from this type of system to satisfy users' needs. In addition, it may have a significant impact on commercial applications by helping users to find relevant items (Lü et al., 2012a).

2.2.3 The aim of recommender systems

The aim of recommender systems is to find a way of helping users by suggesting interesting items based on their interaction and behaviour. The main aim is to recommend items that satisfy the user. To do this, the system requires a way of discovering the most relevant item and, consequently, solving the information overload problem associated with the Web. Recommender systems can filter and rank the top items that meet user preferences. They potentially increase the profits of companies that apply the recommendations (Lü et al., 2012a).

2.2.4 The importance of recommender systems

The primary function of recommender systems is to locate items that are relevant to the user's information needs, but they can also be used to check the importance of a web page (looking at the position of the page in the results list of a query) or to discover the various uses of a word in a collection of documents (Adomavicius et al., 2015). The key importance of recommender systems is as follows:

1 Recommender systems play an important role in solving the 'information overload' problem. This is especially important given the rapid growth of information. The software used to develop solutions to this problem incorporates techniques from the fields of information retrieval and information filtering.

2 Personalisation: tailoring user needs and filtering items to make suggestions based on the users' actions.

2.2.5 Major challenges of recommender systems

2.2.5.1 Sparsity

It is widely acknowledged that data sparsity poses difficulties in recommender systems but there is no consensus regarding the precise definition of the term. It is associated with dataset representation. One definition of sparsity is when a user offers only relatively few ratings. Potential solutions to the sparsity problem include an algorithm that computes the similarity between users and items, but the ratio of the number of items utilised in training and testing differs. Both CF and CBF suffer from the sparsity issue. For example, when CF is applied the similarity between two users is calculated based on common rated items. In cases where users have no common items that are rated with others a calculation based on closest neighbour may be misleading, which affects the accuracy of the recommendation. Similarly, CBF has some limitations in selecting items for users with very little content in their profile.

Several methods have been proposed to alleviate the sparsity problem. These methods need to consider either reducing the dimensionality of the user CF interaction matrix or utilising extra information such as demographic attributes like age, gender and occupation (Zhou & Luo, 2010). Melville & R. (2002) proposed a content-boosted CF method that combines content information with CF to reduce the missing values in a user-item matrix. However, this method was not good enough for users with a small number of ratings. Ma, King and Lyu (2007) presented a user-based CF method that modifies the traditional Pearson similarity measure using weighted factors through common rated items. Another method, that utilises associative retrieval to find the relationship between users using a graph

model, was developed by Chen, Wu, Xie and Guo (2011). This method computes the similarity between users using direct and indirect similarity distance. However, this method increases the complexity of the data processing over time.

It is clear that the sparsity problem has not yet been solved using a simple and straightforward method with no additional information needing to be added to make up for the missing values. The main aim of this thesis is to find a way to alleviate the sparsity issue using a hybrid method and by utilising a modified similarity measure that applies the number of ratings a user has made and a common rating between two users.

2.2.5.2 Scalability

Scalability is about growth in the number of users and items. For example, on Twitter there are a lot of users who explore the website daily and who share a large amount of information. This type of application requires a high level of recommendation to scale users' interests and preferences (Thorat, Goudar & Barve, 2015). To address such scalability, George (2005) proposed a collaborative filter based on co-clustering method that generates predictions from a user's nearest neighbour using clusters of particular users and items.

2.2.5.3 Diversity

Recommender systems are expected to increase diversity by recommending unknown items or new products. However, due to the growth in information and the limitations in presenting a single algorithm, there may be a problem with recommendation accuracy. To overcome this problem, a hybrid approach is proposed here to enhance the efficiency of recommendations (Thorat et al., 2015).

2.2.5.4 Cold start problem

The cold start problem occurs due to the fact that a new user or a new item with no associated knowledge has been discovered. For example, when a user uses a system for the first time, there is difficulty in making a recommendation, because there is not sufficient information to identify their interests. Despite the research that has attempted to solve this issue, it is still one of the limitations that recommender systems face (Ahn, 2008; Lam, Vu & Le, 2008; (Tandberg & Øystein, 2015). New items in a system also pose the same problem. However, the new item's features may help in finding similar items. To tackle this problem, a hybrid method is an option, which can help by giving additional information that fits the missing values. For example, a new similarity measure is proposed by Ahn (2008) which improves the cold start condition using three factor measures, proximity, impact and popularity, to calculate the similarity between two users.

2.3 User profile

The most important elements in producing recommendations are users and items. Users are represented in such a way that the recommender model can get their interests and preferences. Item descriptions are used to accurately recommend relevant items to users. Hence, the main steps of a recommender system are identifying the target user and the item that is recommended.

A user profile is a collection of information about an individual user that reflects the user's interests and preferences. It also plays an important role in obtaining knowledge about users of software applications. Examples of applications include intelligent tutoring systems and adaptive educational systems. In the context of users of software applications, a user profile or user model contains essential

information about an individual user (Amo et al., 2015). The motivation for building user profiles is that users differ in their preferences, interests, background and goals. Discovering these differences is vital for providing users with personalised services. The user profile has become a key area in the development of personalisation systems (Godoy & Amandi, 2005). It can be used to store a description of the characteristics of a person (Sharma, Sharma & Gupta, 2012). This creates an opportunity for a recommender system to acquire greater insight into the interests of a user, and this can be utilised at a later date for the purposes of a reasoning-based recommendation process (Blanco-Fernández et al., 2008).

In order for the personalisation process to be successful, the user profile that is created must be highly accurate. Creating a user profile typically involves a two-stage process: profile learning and profile representation (although profile learning is not at all times strictly necessary). Furthermore, profiles may be either static or dynamic, meaning that they evolve as users' interests change. Intelligent user profiles are compiled using a variety of techniques, including demographics, weighted semantic networks, classifiers (CBR, Bayesian networks, decision trees etc.) and hybrid combinations. They are utilised in recommender systems, knowledge management systems, adaptive systems and intelligent agents.

The user profile is a determining element in accessing pertinent information during a search session. The way the user profile is adapted enables personalised access to the contents of relevant documents (Berisha-boh, 2006). Personalised Internet searches are more successful if the user's interests and preferences are known. Therefore, in order to offer a truly personal Internet search, the creation of a user profile is highly beneficial. Such a profile effectively models the needs of a user

according to insight provided by their Internet usage patterns (Rakesh Kumar & Sharan, 2014).

	Explicit	Implicit
User	Ratings (Gipp, Beel, & Hentschel, 2009) Like/dislike (Bogers & Van Den Bosch, 2009) Reviews (Koren, 2008) Surveys (Gauch et al., 2007)	Click through rate (CTR) (Moawad, Talha, Hosny, & Hashim, 2012) Eye tracking (lab) (Hannak et al., 2013) Time tracking (Nanda, Omanwar, & Deshpande, 2014)
Item	Ratings (Gipp et al., 2009) Product descriptions (H. Wang, Wang, & Yeung, 2014) Reviews (S. Zhang, Yao, & Sun, 2017)	Ratings (Gipp et al., 2009) Keywords/products (Nanda et al., 2014) Categories (Nanda et al., 2014)

Table 1: Information type and method of data collection.

2.3.1 User information

A collection of user information enables the recommender system to identify the user's needs. For example, this could consist of user ratings that represent the user's preferences and demographic information. This information is divided into either explicit or implicit data. Explicit data includes user information collected from the users inputting this information. For instance, explicit ratings involve asking users

to rate items on a nominal scale (e.g. 1 to 5). Demographic information such as age, gender and occupation are categorised as explicit information.

In recommender systems, the user profile is widely applied to explicit data to suggest interesting items. For example, Netflix generates movie recommendations based on items rated by users in the past.

With regards to explicit information there are some drawbacks that need to be considered. First, some users are unwilling to provide such information to help the system. Secondly, privacy concerns mean that some users are not confident about sharing their information. These problems adversely affect the accuracy of recommendations and make it difficult to find or suggest appropriate items.

In contrast, implicit information is obtained from users by tracking the user's interaction with the system. It is primarily based on the user's behaviour; for instance, when user is browsing the Web their viewing history is saved. Another example is that purchases are recorded each time a user is shopping in an online shop. Implicit information can easily be gleaned from a user's browsing history (Liu, Yu & Meng, 2004). It utilises insights drawn from each web page that a user has visited in order to better understand their real-time preferences. Ziegler, McNee, Konstan and Lausen (2005) advocate the use of item taxonomy to indicate the topic interests of users and issue recommendations for specific items. Kim and Chan (2006) argue that users' interests can be interpreted using the concept-hierarchical approach by compiling keywords or topics derived from their browsing histories.

Users may have very diverse goals and characteristics. In order to personalise recommendations, recommender systems exploit a range of information about the

users. The system must then adapt to the changes made over time by the user and updating their user profile accordingly.

2.3.2 Item information

Item information represents the item details. It can also be derived either explicitly or implicitly. For example, the type of movies that the user has rated in the past provides an implicit indication of the item categories that this particular user is interested in. Therefore, it helps to find similarity between users that have rated similar items. Some work has applied knowledge bases to represent items in more detail. Anand, Kearney and Shapcott (2007) created an item ontology to represent movies. Each movie is described using various attributes such as director, actors and running time. The attributes may include sets of objects as well as being standard numeric or string objects. This ontology is used to calculate whether or not certain attributes of a movie have an impact on user preferences.

Objects that are recommended are referred to as items and these items can be classified in terms of their value, complexity or utility. An item can have either a positive or a negative value. An item with a positive value will be useful to the user, whereas an item with a negative value would not be suitable.

The process of acquiring an item involves a cost that is not only the financial cost necessary to complete the purchase but also a cognitive cost in terms of the resources consumed in searching for the item. Therefore, when designing a new recommender system, it is necessary to make allowance for an item's textual presentation and structure as well as the time-dependent nature of the item. Shenghui Wang et al. (2014) enhanced content-based recommendation using explicit item relations, linking them together to semantically enrich each item, in

order to find related concept more easily and produce more accurate recommendations. The designer of a recommender system must be aware that even if the user is not paying a financial fee to access an item, they are still incurring a cognitive cost by searching for and consuming an item. However, if the item selected is useful to the user, it is highly probable that the benefit far outweighs this cost. Conversely, if the item is not useful to the user, the net value of the experience for the user is negative. Be that as it may, for items such as motor vehicles or financial investments, the financial cost associated with items is the overriding factor to bear in mind when deciding upon the best-suited methodology (Adomavicius et al., 2015).

2.4 Personalisation

The objective of personalisation for the purpose of delivering personalised information is fairly straightforward. It is to deliver information that is relevant to an individual or a group of individuals in the format and layout specified and within the time intervals specified. When information sources are updated, it is important that the updated information is delivered to individuals. Updated information may be delivered immediately upon updates to the information sources or based on a schedule specified by individuals or by a system default (e.g. once a day, once a week, once a month). Successful personalisation of data access presents two fundamental difficulties: precisely recognising the user connection and sorting the data in such a way that matches the specific connection (W. Kim & Solutions, 2002).

Efforts to deliver personalisation usually require information about a user's interests to be collected in order to compile a user profile. This information can be amassed either implicitly or explicitly (Fathy, 2014).

The advantage of personalised recommender systems is that they are responsive to the preferences of users. While it is true that all recommender systems respond to the preferences of users, not all recommender systems are personalised. For instance, conversational recommender systems are responsive to information gleaned from users in sessions, thereby ensuring a degree of personalisation when making recommendations. However, this example would be classified as weak personalisation because it is based solely on the information provided in the session. As such, the recommender is unable to respond to preferences unless they are expressed within the session. If two users respond in similar ways during a session, the recommendations made to these users will be alike. However, in reality, these two users may have significantly different preferences over the long term. It is therefore apparent that if a recommender system is to achieve a high degree of personalisation, it must have access to persistent user profiles that can be utilised in conjunction with in-session feedback to enhance the recommendations issued.

Predictive personalisation is defined as the ability to predict customer purchase, needs or wants, and precisely tailored to offer communications accordingly. Social data is one source for this predictive analysis, particularly social data that is structured. Predictive personalisation is a much more recent means of personalisation and can be used to augment current personalisation offerings (Q. Wang & Jin, 2010).

The presentation of a website's content can be tailored to match a specific user's instructions or preferences. This custom tailoring is accomplished either by the user choosing from a menu of available options or by tracking his or her behaviour on the site (such as which pages are accessed and how often) (Mobasher, 2007). Callan and Smeaton (2003) focus on personalised recommendations in the domain of digital libraries and how the personalised system can help to assist a user in navigating a large body of online information. Huang, Huang and Chen (2007) propose a personalised e-learning system using genetic algorithms with a CBR system to construct an optimal path for each learner.

Fathy (2014) studied the way in which user profiles enhance the re-ranking of search results and contribute to more pertinent information on the Internet. Semantic user general and specific interests are combined to enable re-ranking and enhance the quality of searches, relative to traditional searches. (L. Li, Yang, Wang, & Kitsuregawa, 2007) advocated a new dynamic user profile that reflects changing interests over time and re-ranks search results accordingly. In addition, Kumar and Sharan (2014) proposed an enhanced user profile framework that relies on exploiting information about a user's browsing history supplemented with domain knowledge.

The above review of the empirical literature demonstrates the importance of a well-defined user profile, necessary in order to identify items of particular interest or relevance. In addition, the literature emphasises that recommender systems are reliant on the user's own representation in order to identify what may be of interest to them. As such, the user profile is correlated with the quality and accuracy of

recommendations issued; more knowledge about the user will result in recommender systems becoming more accurate.

2.5 Recommendation methods

Recommendation systems use a number of different methods to solve the recommendation problem. Hence, details of the recommendations methods that have been used across the all experiments are provided. These methods can be implemented based on domain requirements and are able to identify and predict items that meet the user's interests. The methods utilise different recommendation algorithms to make suggestions and recommendations based on the saved interaction information between users and items. Recommendations can be provided through rating interaction; this is called collaborative filtering (CF). Another method uses an item's description to filter and predict the similar items; this is called content-based filtering (CBF). In addition, case-based reasoning plays an important role in recommender systems and has been successfully applied to build a user model and re-use previous cases to make recommendations (Bridge, Goker, McGinty & Smyth, 2005). Demographic filtering is a method that applies demographic attributes to find what kind of users rate a specific item. A hybrid method combines two or more methods to improve recommendations (Burke, 2007). The following sections explain each method in detail.

2.5.1 Collaborative filtering

Collaborative filtering (CF) is considered to be the most popular technique for recommender systems. It has been widely implemented in different domains to make recommendations. It is a method of information filtering that seeks to predict the rating that a user would give to a particular item based on a similarity matrix.

CF provided the foundation for the first recommender systems and these systems were used to 'help people make choices based on the opinions of other people' (Goldberg, Nichols, Oki & Terry, 1992). It helps users to find relevant items and makes suggestions based on similar users' tastes. It has been applied in a variety of areas and application such as movies, books and research articles. In this approach the similarity calculation is based on the user's peers.

The task is to make an automatic prediction by considering similar users' ratings for an item. The basic idea of CF is to find users whose past rating behaviour is similar to the current user, and the algorithm tries to predict this. This approach uses the k-NN algorithm to calculate recommendations and the main data required are the rating matrix and the similarity function that computes similarity between users.

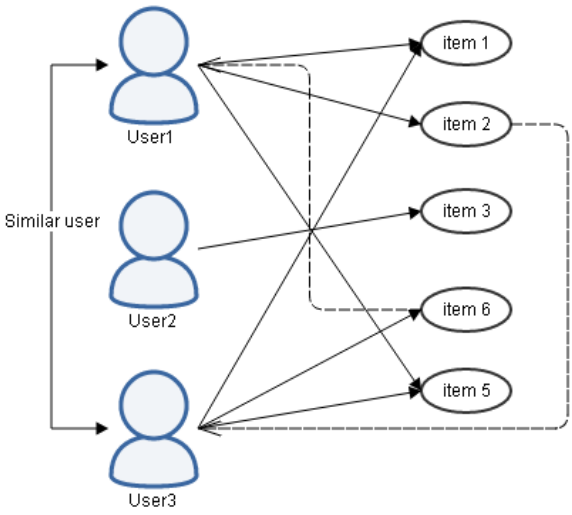

Figure 1: User-based collaborative filtering

Figure 1 illustrates user-based collaborative filtering used to compute similarity. User 1 has rated items 1, 2 and 5. User 3 has also rated items 1, 5 and 6. In this scenario, there is similarity between User 1 and User 3, so Item 6 is recommended

to User 1. Meanwhile, Item 2 is also recommended to User 3. By contrast, User 2 has no similar items with both users, which there is no recommendations has been made to this user.

The similarity in the tastes of the three users is calculated based on the similarity in the rating history of each user. This is called user-to-user correlation. Being a member of a community confers benefits for an individual in terms of access to a wider knowledge base and more diverse experiences. This insight can then help to influence the opinions of the individual or help them to make decisions based on items that have been rated. CF systems are used by individuals in order to find items that are likely to be of interest to them, to receive recommendations from other users or to interact with other like-minded community members (Spiegel, 2009). Hannon, McCarthy and Smyth (2011) applied CF to find similar users based on the follows ID, follower ID or both.

Previous ratings are fed into the algorithms so that intelligent rating predictions can be produced that are based on learned models. Data mining techniques and machine learning algorithms help to develop the models and provide insight into rating patterns. Memory-based CF algorithms are subject to heuristic prediction rules; this is not the case with model-based approaches. Examples of model-based CF algorithms include linear regression models, Bayesian network-based models, matrix factorisation (MF) models, latent factor models and singular value decomposition models (SVD). Importantly, prediction performance is significantly enhanced by using model-based methods (Adomavicius et al., 2015).

Over time, MF and latent factor models have become more popular in applications involving recommender algorithms for both implicit and explicit feedback. MF

categorises users and items in terms of factor vectors that are derived from item rating patterns. There are many examples of empirical studies that have employed MF and neighbourhood methods to enhance the performance of collaborative filtering and lessen the adverse effects of the cold start problem (Spiegel, 2009).

2.5.1.1 User-based CF

User-based CF is widely used as a baseline approach. This method looks for similarity between users based on rating patterns (Spiegel, 2009). It makes a recommendation based on the similarity between the target user and other users. The most popular similarity methods are the neighbourhood model and latent factor models. The neighbourhood method is the most common method used in baseline CF. The idea is that, for a given user, the preferences of similar users (neighbours) can serve as recommendations. Amatriain, Lathia, Pujol, Kwak and Oliver (2009) proposed a user-user approach as an appropriate method for recommending items based on expert opinions. Another example is provided by Bay, Kong, Building and District (2010), where mobile activities were recommended to users based on other locations.

2.5.1.2 Item-based CF

Item-based CF predicts recommended items based on similarities between items shared with similar users and calculates the similarity with the target item (Spiegel, 2009). Linden et al. (2003) designed an item-to-item collaborative filtering approach that matches the items rated or purchased by the user with other, similar items.

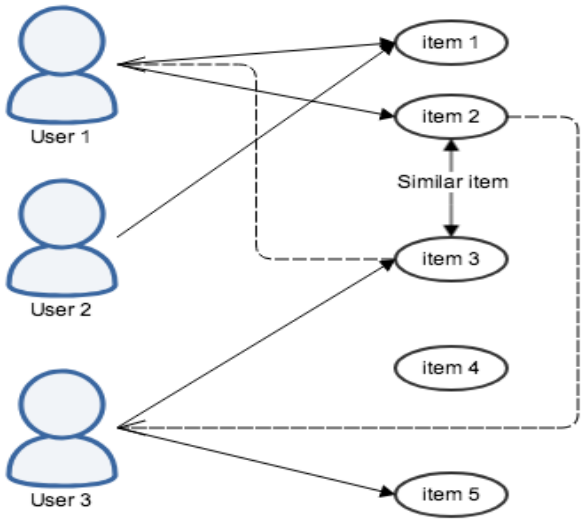

Figure 2 Item based collaborative filtering

As it can be seen in Figure 2, item based CF calculates the similarity between items through considering the set of items that a specific user has rated and computes how similar to the target items. For example, item2 and item3 are similar and user1 has rated item2, hence item3 is recommended to user1.

2.5.1.3 Advantages of CF

1. CF techniques make implementation of recommender systems easier.

2. CF can improve prediction performance.

2.5.1.4 Major challenges in CF

CF faces several challenges and limitations such as sparsity, scalability, diversity and the cold start problem. These limitations affect the quality of recommendations and thus CF requires additional algorithms to overcome these challenges.

The sparsity problem stems from having a large number of items in the dataset. Users are not provided with enough ratings for these items. As a result, in the

empirical literature a lot of researchers have proposed using CF with other methods to reduce these problems (e.g. L. Zhang, Tao & Teng, 2014).

Clustering algorithms based on collaborative filtering markedly enhances the accuracy of recommendations and helps to address the problem of sparse data. Sun, Wang and Guo (2009) further improve the quality of predictions by applying user preferences and descriptions of the items. A probabilistic relational model drawing upon CF is employed, thereby enhancing the quality of predictions and addressing the problem of sparse data.

Scalability becomes a problem when the number of users and/or items grows significantly. This will remain a problem if CF is used on its own, but combining CF with other algorithms provides a viable solution. Sun et al. (2009) advocate combining the descriptions of items with the preferences of users. Laveti, Ch, Pal and Babu (2016) introduce a recommender system based on hybrid similarity metrics that simultaneously utilises several calculation functions using weighted methods.

Diversity becomes a problem when there are more items than users. In the movie dataset explored in this study, 6,040 users have given more than one million ratings for 3,900 movies. In this scenario, CF would be prone to sparsity because it is possible that users have not rated similar items, resulting in low-quality predictions being made. Ziegler et al. (2005) use intra-list similarity metrics in a bid to enhance topic diversification in a recommendation list. The intra-similarity metrics allow for specific details of an item to be included, such as the actors featuring in the movie, the genre of the movie etc.

The cold start problem arises when a new user or new item is added to the system. At this early stage it is especially difficult to be aware of their preferences in order to arrive at suitable recommendations. Recognising this problem, numerous attempts have been made to combine different methodologies in order to lessen the negative effects of this situation (e.g. Duzen & Aktas, 2016; Cao, Ni & Zhai, 2015). Duzen and Aktas (2016) devised a methodology that combines the CF approach with an ontology-based CBR method.

2.5.2 Content-based filtering

CBF approaches utilise user profiles and the content of items as domain knowledge, and compare information from new items with the user's profile (Wanvimol Nadee, 2016). The general principle of content-based methods is to identify common characteristics of items that have received a favourable rating from users and then recommend new items to users that share similar features (Adomavicius et al., 2015; Ziegler, McNee, Konstan & Lausen, 2005). For example, when the generation of recommendations is based on content, items can be retrieved and filtered based on specific search queries entered by the user or keywords that describe previously purchased items (Redpath, 2010). The users' interests or preferences can be described in terms of their interest in item characteristics such as topics, attributes or categories (Wanvimol Nadee, 2016).

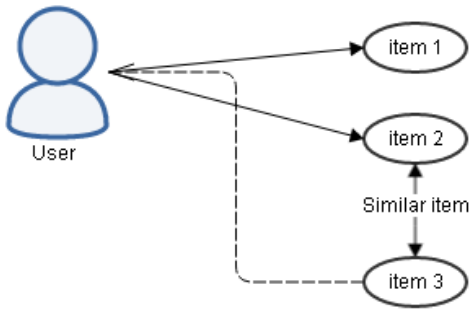

Figure 3: Content-based filtering

In CBF, an item is recommended based on the properties of that item. For example, in the MovieLens dataset, if a user has rated several action movies then recommendations will be made based on movies classified in the dataset as belonging to the 'action' genre. As shown in Figure 2, Item 3 is recommended to the user because of its similarity with Item 2, which the user had previous liked/rated.

2.5.2.1 Related work

Most CBF approaches rely on information retrieval techniques (Pazzani & Billsus, 2007). Hannon, Bennett and Smyth (2010) present a content-based recommender system based on the words in each tweet utilised by a user. Usually, the description of each item is represented in each row by a set of values. Depending on the domain, the representation could be provided as a Boolean value or as an integer. The information retrieval count is based on the number of times that a particular word appears in a document or web page. This is achieved by text analysis methods and the most popular methods are divided into two approaches: heuristic-based and model-based. For instance, in heuristic-based approaches, term weights are

employed which calculate the relevance of the documents by considering term frequency (TF) and inverse document frequency (IDF). Hence, the profile of the user is represented as a vector of weight for each feature.

In model-based approaches, the user's preferences are calculated by different techniques through probabilistic methods, machine learning and linear classifiers. Typically, a user's profile is represented by a description of the item that they are interested in or have interacted with in a recommender system. The history of the user's interactions is collected either by implicit or explicit methods. Implicit information is generated from the items that a user visits, likes or rates. It reflects the user's behaviour and can assist the recommender system to recommend or rank items that are most likely to match the user's preferences. Moreover, it can more effectively help model the user in order to train the machine learning and data mining techniques. (Burke, 2002) presents a personalised recommendation system that learns from user feedback and compares a user profile with a document to make a recommendation. However, this suffered because of the ambiguity of the terms, which led to recommendations of irrelevant documents. Hannon et al. (2011) find similar users that frequently tweet specific terms from either the users' tweets or their followers' tweets.

In summary, CBF assumes that similar items share similar objective features that describe their content. The challenge in this method is to predict and match these features with other similar items. For instance, when recommending movies, the genre is the most important feature for the content.

2.5.2.2 Advantages of CBF

1 Content-based recommender systems provide users with independence through exclusive ratings that are used by the active user to build their own profile.

2 Content-based recommender systems provide transparency to active users by giving explanations of how the recommender system works.

3 Content-based recommender systems are capable of recommending items not yet placed by any user. This will be advantageous for new users.

2.5.2.3 Major challenges in content-based filtering

The limitations of this method occur when item features are limited, which results in inappropriate recommendations being made. It also suffers from over-specialisation, which occurs when recommended items are very similar to items that the user has already rated. These limitations can be outlined as follows:

1. In some cases it is problematical to generate attributes for specific items.

2. CBF is prone to overspecialisation because it tends to recommend the same types of items.

3. When using CBF it is more difficult to obtain feedback from users because they are unlikely to rank items (unlike in the CF approach). Consequently, it cannot be discerned whether or not the recommendation was appropriate.

CBF also suffers from the new user problem. A user who has only rated relatively few items will not be offered reliable recommendations. In order to present relevant

recommendations, a user should rate enough items to help a content-based system to understand the user's interests and match these with other items not seen by this user. Therefore, in an absence of sufficient item ratings, the accuracy of the recommendations will be adversely affected for new users.

2.6 Demographic filtering

It is possible to identify the type of person that likes a particular item by referencing their demographic details (Pazzani & Billsus, 2007). User attributes are incorporated into demographic recommender systems and this demographic data is used as the basis for arriving at suitable recommendations, sometimes relying on pre-generated demographic clusters (Vozalis & Margaritis, 2004). This information is gathered either explicitly through the user's registration or implicitly via their navigation of the system they use (Moldovan & Muntean, 2009). Subsequently, demographically similar users are identified by means of the recommendation algorithm. Recommendations are based on how similar people (in terms of their demographics) rated a particular item (N Tintarev, 2009). (Vozalis & Margaritis, 2004) present a hybrid algorithm that keeps the core ideas of two existing recommender systems and enhances them with relevant information extracted from demographic data. Pazzani and Billsus (2007) present an approach that considers user profiles as vectors constructed from demographic attributes such as age, gender or postcode to find relationships with other users and calculate similarities between users in order to generate the final prediction.

Demographic-based filters are similar to collaborative filters in the sense that both are able to identify similarities between users. In this case, demographic features are used to determine similarity rather than their previous ratings of items (N Tintarev,

2009). Mittal (2014) proposed that demographic attributes should be added as metadata to help the neighbourhood algorithm find similar users. He shows the importance of this metadata in presenting significant results and providing better recommendations.

Redpath (2010) states that demographic information helps to address the cold start problem. This is because this approach doesn't require a detailed history of user ratings before making recommendations, unlike the content-based and collaborative approaches (Burke, 2007). (Redpath, 2010) also explains the association between collaborative filtering and the demographic base as a good way of combining them to enrich user preferences and more accurately identify their interests. Gupta (2015) proposes a combination algorithm that clusters users based on demographic attributes using a weighed scheme. It solves the cold start problem by assigning a new user to the nearest cluster using demographic similarity.

2.7 Hybrid method

Hybrid approaches involve utilisation of two or more of the recommendation approaches. They can overcome the limitations and the challenges involved in relying on just one recommendation approach. Commonly, collaborative filtering and content-based recommendation approaches are combined to tackle the problems associated with each algorithm individually. However, knowledge-based systems also offer good recommendation methods in combination with others because they can obtain certain knowledge about either the item or the user to be compared in a case base. The hybrid recommender system approach seems to solve the deficiencies of CF and CBF in several recommendation applications. It is for this reason that the literature shows an increased number of applications of hybrid

approaches over the years. Furthermore, there seems to be an optimisation standard in matching the shortcomings of the main recommendation techniques (J. Sun, Zhao, Antony & Chen, 2015). The key advantage of this method is that it generates better quality recommendations and avoids issues such as sparsity, cold start and the long tail.

The purpose of combining two or more methods is that each one has its own limitations that adversely affect the quality and performance of recommendations; e.g. a lack of information about the users or items. Burke (2007) proposed a hybrid recommendation techniques and possible ways of presenting the combination with the following classes:

Weighted

This taxonomy combines the score of recommendation techniques using a linear formula. The scores are combined to provide a single recommendation using methods such as linear combinations or voting schemes (Pazzani, 1999).

Switching

This hybrid method involves selecting and switching between recommendation techniques using certain criteria. For example, daily learner is a switching hybrid user model that represents an adaptive news recommendation by identifying long-term or short-term user preferences. It switches between content and collaborative filtering in which content-based is applied first. If CBF is not able to make a suitable recommendation, CF is then applied (Billsus & Pazzani, 2002).

Mixed

Recommendations from different recommender techniques are employed at the same time. This is suitable in cases where the recommendations need to be presented together. For example, Glauber, Loula and Rocha-Junior (2013) propose a mixed hybrid recommender approach for a given name using more than one technique. This approach avoided the cold start problem for new items because in CBF the recommendation is based on the similarity of the description of the items, even they have not previously been rated by a user.

Feature combination

This method involves merging the features of CF as a simple component and then using the other techniques' features. In this scenario, CF is considered without completely relying on it. Hence, it reduces the sensitivity of the number of users who have rated an item. By contrast, it allows the system to use the content of the information for items that are related to CF.

Cascade

The cascade hybrid approach involves a process of refining the recommendations produced by the second technique as a 'tie breaker'. Furthermore, the system avoids any items that have already been eliminated by the first technique. The initial step is to apply a recommendation technique such as CF in order to compile a crude ranking system. This ranking is then refined.

Feature augmentation

In many ways, this class is akin to feature combination hybrids, but where they differ is that novel characteristics are produced by the contributor and it is more flexible than the feature combination approach (Thorat et al., 2015).

Meta-level

This approach uses the output of the first method such as CF as an input for the second method like CBF. In the first method there is an initial recommender component and this is unlike the feature augmentation approach because the raw data are completely replaced by the learned model (Martin-vicente, Gil-Solla, Ramos-Cabrer, Blanco-Fernandez & Lopez-Nores, 2010).

Im and Park (2007) proposed a hybrid system that applies CBR and neural networks to improve the retrieval accuracy of CBR by training the feature weighted using neural network (NN), which leads to improvements in prediction accuracy compared to the pure k-NN. Meanwhile, Blanco-Fernández et al. (2008) apply a reasoning hybrid approach that employs a semantic process in the content-based approach as a first phase to determine a target item, then CF is used to discover more about user relation to find the similar users. A hybrid framework was also proposed by Bremer, Schelten, Lohmann and Kleinsteuber (2017) that utilised a collaborative filter relying on user/item metadata and demographic data. The framework benefits from the similarity between users via a correlation in terms of demographic attributes. This leads to improvements in prediction accuracy and is able to solve the cold start problem compared with the baseline method. The authors point out the importance of collecting item metadata in overcoming the challenge in which users and items have little information.

Among these seven hybridisation techniques, four of them – weighted, switching, mixed and feature combination – are order independent, which means there is flexibility in terms of the order in which they are applied. In other words, a combination of CBF and CF is the same as a combination of CF and CBF. By contrast, the other methods are order sensitive, which means the results might be different if the starting method is changed (Pradesh, 2018).

The challenge in hybrid recommender systems is how to incorporate both CF and CBF techniques together, resulting in high-quality recommendations. This research presents an approach based on a switching hybrid system and a multi-level method in order to overcome the sparsity problem, mainly the long tail. It aims to improve the accuracy of predictions using two techniques according to one of the hybridisation methods represented above. The details of this approach are provided in Chapter 3.

2.8 Case-based reasoning

A CBR recommender system adopts solutions based on the previous experience of similar cases. It is assumed that similar problems will have similar solutions (Kolodner, 1992). It is a knowledge-based system that stores past cases to handle new cases. Each case represents a set of queries and solutions for these queries. It is a reasoning cycle described by the four main CBR steps, as illustrated in Figure 4 below.

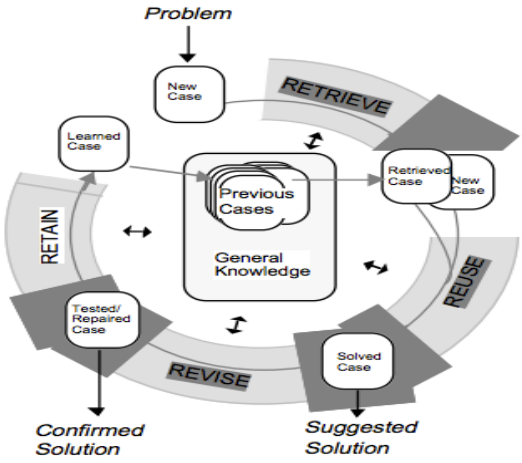

Figure 4: CBR cycle (Aamodt & Plaza, 1994)

Case-based recommenders operate on the principals of similarity and retrieval. Each item is represented as a case and recommendations are made by selecting items that are deemed to have the closest similarity to what the user has requested or the closest similarity to their profile (Smyth, 2007).

2.8.1 The CBR Principles

To further understand the CBR cycle, here the four steps are described in more detail:

Retrieve refers to finding similar cases from the case library. It can be applied using different similarity methods such as nearest neighbour, inductive indexing and knowledge guidance. However, the nearest neighbour retrieval method is the most widely used (Li, Sun & Zhangi, 2015). This step is highly important because cases are recalled to find an appropriate solution (Kolodner, 1992).

Reuse refers to a solution suggested by a similar case.

Revise is where a solution is adapted to better fit the new problem, if necessary.

Retain is the new solution once it has been confirmed or validated.

CBR is becoming an important approach in domains with available cases. It uses the similarity function to determine the best matched cases from the case library, those that are associated with the current case. Therefore, the quality of CBR solution is depending on the previous experiences that is saved and the ability to match and adapt a new case (Kolodner, 1992). The aim of using CBR is to utilise the knowledge-based method to retrieve past experiences and adapt them to make a recommendation (Bridge et al., 2005). CBR systems begin their reasoning from cases that are associated with the knowledge base. Hence, it is a capable system that can learn and automatically assign a solution based on previous behaviour (Bichindaritz, 2015).

When items can be clearly defined in terms of their features (colour, size, price etc.), CBR can be successfully applied in order to make accurate recommendations. Case-based recommenders effectively make a series of judgements regarding the similarities between products so as to make better quality recommendations. Therefore, this approach is well-suited to many online retailers, especially if the preferences of users are not clearly defined. The following sections outline a suitable methodology for delivering case-based recommendations, emphasising the differences between this and the alternative approaches that are available (Smyth, 2007).

2.8.2 Related work

Case-based reasoning appears to be a popular technique in creating recommender system algorithms. For example, Gedikli, Jannach and Ge (2014) detail how online stores have used past information via CBR in order to increase sales of books,

movies, mobile phones and other devices. In addition, CBR is used with other technique to improve the prediction accuracy. For instance, Im and Park (2007) propose feature weighting CBR as recommender system with neural networks to predict customer characteristics and thus market behaviour, in which improves the prediction accuracy. Furthermore, Cao et al. (2015) combined a CBR system with CF to solve the cold start problem by retrieving the source case to find a solution to the target case.

On the other hand, Case-based reasoning personalised recommendation systems provide a link between users and information services systems. As such, they help the transmission between demand and services by means of a platform for information exchange and the processing of data (F. Li et al., 2015).

Therefore, the challenge in a CBR system is to build a knowledge library, because a case that could cover a set of problems that arise in a certain domain (Kolodner, 1992).

To sum up, among the various hybrid approaches within the empirical literature, solutions that use CBR and CF methods together have received most attention. The hybrid recommender system proposed by Cotter and Smyth (2000) combines CF and CBR approaches based on the favourite channel that a user selected in the past, their fields of interest and the watching time (Duzen & Aktas, 2016).

2.9 Multi-level method

A multi-level method was developed by Polatidis and Georgiadis (2016), to improve the similarity between users that is found in a CF method, thus addressing the limitations that are related to the traditional measures used by CF. Most CF

approaches analyse user ratings to determine the similarity between users and items. The similarity measure is important for finding accurate results in recommender systems. However, it is challenging to determine distance measures in these systems in order to find similarities between users. Collaborative filtering is the most commonly applied algorithm through the k-NN approach (Jeong, Lee & Cho, 2010). The key issue in this technique is how to calculate the similarity between users or items by finding similar shared interests. It significantly relies on the rating aspect, which allows users to assign a high or low rating to a certain item based on their preference or dislike for it (Konstan & Riedl, 2012). Many similarity measures have been adopted in recommender systems such as Pearson's correlation coefficient (PCC) (Resnick et al., 1994) and cosine (Shi, Larson & Hanjalic, 2014) to provide recommendations based on absolute ratings between users. Modified similarity measures are one of the most important challenges to improving prediction accuracy in recommender systems. The multi-level method has been proposed recently as providing improvements to the accuracy of the recommendations that is based on Pearson similarity (Polatidis & Georgiadis, 2016). It has been utilised in a variety of different applications. The method is used to find similarity between users, provide better quality recommendations and improve accuracy.

$$Sim_{a,b}^{PCC}$$

$$= \begin{cases} Sim_{a,b}^{PCC} + x1, if \ \left|\frac{Ia \cap Ib}{T}\right| \geq t1 \ and \ Sim_{a,b}^{PCC} \geq y \\ Sim_{a,b}^{PCC} + x2, if \ \left|\frac{Ia \cap Ib}{T}\right| < t1 \ and \ \left|\frac{Ia \cap Ib}{T}\right| \geq t2 \ and Sim_{a,b}^{PCC} \geq y \\ Sim_{a,b}^{PCC} + x3, if \ \left|\frac{Ia \cap Ib}{T}\right| < t2 \ and \ \left|\frac{Ia \cap Ib}{T}\right| \geq t3 \ and \ Sim_{a,b}^{PCC} \geq y \\ Sim_{a,b}^{PCC} + x4, if \ \left|\frac{Ia \cap Ib}{T}\right| < t3 \ and \ \left|\frac{Ia \cap Ib}{T}\right| \geq t4 \ and \ Sim_{a,b}^{PCC} \geq y \\ 0, if \ \left|\frac{Ia \cap Ib}{T}\right| < t4 \ \ otherwise \end{cases} \quad (1)$$

Where $Sim_{a,b}^{PCC}$ denotes the similarity between user a and user b. T stands for the total number of co-rated items; $t1, t2, t3$ and $t4$ are the threshold of co-rated items for user similarity. It is considered that $t1 = 50, t2 = 20, t3 = 10$ and $t4 = 5$. In addition, x is defined as $x1 = 0.5, x2 = 0.375, x3 = 0.25, x4 = 0.125$ and $y = 0.33$.

2.10 Multi-criteria

The multi-criteria approach represents users' preferences through considering the attributes of items. For example, in the movie domain the criteria can be the movie genre, actors, director and year. However, the difficulty of this approach is the method of incorporating it with the CF technique and computing similarity. Wasid and Ali (2018) use a clustering method to find similar user profiles using multi-criteria preferences. Manhattan similarity is then used to compute the distance between two users in the same cluster. Fleeson et al. (2017) present a combination method that uses item-based and multi-criteria approaches in a genetic algorithm.

2.11 Prediction algorithms

With prediction tasks, it is usually expected that the most utilised evaluation methods is to measure the accuracy of the prediction. Most recently, the published research on recommender systems has been evaluated using predictive accuracy, relying on the error or the correlation measure (Konstan & Riedl, 2012).

Practically, recommender systems perform prediction algorithms on an unknown user-item ratings matrix that must be calculated in real time. Table 2 and Table 3 provide overall comparison of the most commonly applied algorithms that is implemnted in (Nicolas, 2017). These tables compare the predictive accuracy measuers using MAE, RMSE and the time spent to calculate the rating. This comparison was conducted using two popular datasets, MovieLens 100K and MovieLens 1M (Harper & Konstan, 2015a), that were also used in this thesis. The following tables show various algorithms with their error rate using two datasets.

The algorithms	RMSE	MAE	Time
SVD	0.934	0.737	0:00:11
SVD++	0.92	0.722	0:09:03
Centred k-NN	0.951	0.749	0:00:10
k-NN Baseline	0.931	0.733	0:00:12
Co-Clustering	0.963	0.753	0:00:03

Table 2: Benchmark prediction algorithms using 100k MovieLens dataset

It can be seen that the baseline k-NN outperforms all the other algorithms in terms of prediction accuracy and performance. Furthermore, the time is significantly

different from the closest algorithm, which is SVD++, which is slightly better in terms of the accuracy.

The algorithms	RMSE	MAE	Time
SVD	0.873	0.686	0:02:13
SVD++	0.862	0.673	2:54:19
Centred k-NN	0.929	0.738	0:05:43
k-NN Baseline	0.895	0.706	0:00:31
Co-Clustering	1.504	1.206	0:00:19

Table 3: Benchmark prediction algorithms using 1M MovieLens dataset

The two most successfully models are k-NN and matrix factorisation (MF), that are utilised to analyse user patterns and find relevant items that match their preferences. However, in the literature there is at all times an argument around the benchmark to compare the proposed method and determine how accurate is it regarding the existing methods.

The experiment was executed on five-fold cross-validation. The folds are utilised in each algorithm using an open source python library called Surprise. All experiments are run on a notebook with Intel Core i5 (2.5 GHz) and 8G RAM. Based on the results above, k-NN is used as a benchmark to the proposed method in this thesis, as explained in more detail in 3.4.1.

Matrix factorisation (MF) is a method that characterises the user and the items by latent factors using the rating detection. The advantage of this method is that it allows the system to utilise additional information to fill in the missing values from the actual data. For example, it can use the explicit feedback from the user to cover

the ratings that are missed in some items, to infer whether this item will meet the user's requirement or not. However, it has more complexity in its modelling compared with neighbourhood models (Koren, Bell, & Volinsky, 2009).

The two most well-known models are SVD and SVD++, described below, which deal with integrating user feedback as a factor model in order to learn more about the user.

SVD is the basic model of MF that maps both users and items to a joint latent factor that explain the correlation between them and obtains a rating based on this dimension. For example, in movie applications, features such as genre are used to measure whether this kind of movie might interest a particular user. This model is used to reduce the sparsity in the user-item matrix (Redpath, 2010).

SVD++ is an extension of SVD that is based on MF model, using implicit rating and feedback information. It utilises the implicit information to identify the user activity and knowledge that can help knowing more about users interest (Rajeev Kumar, 2014).

Centred k-NN is a basic collaborative filtering algorithm that takes into account the mean ratings of each user.

Co-clustering is a collaborative filtering algorithm based on co-clustering. Basically, users and items are assigned to pre-defined clusters. Then, the prediction for an individual user is calculated using other similar users in the same cluster. It is applicable to the large data set that can produce the recommendation with less time (Sarwar, Karypis, Konstan, & Riedl, 2002). It can be seen in table 2 and 3, this algorithm is the fastest one compared to others.

To sum up, it is always a difficult task in the recommender system to decide which benchmark or baseline algorithms to use to make a balanced choice between the more accurate prediction algorithms and the best runtime. However, k-NN was utilised in all proposed algorithms and considered as a benchmark to make a reliable comparison due to the simplicity and straightforward to implement and the efficiency. Therefore, the all conducted algorithms were used by k-NN.

2.12 The long tail recommendation problem

The long tail is when only a small number of ratings has been received for a particular item. In other words, items that are less popular in terms of ratings count. These are known as ‘niches’, and popular items are called ‘hits’. Since an item has received only a few ratings it causes a sparsity issue in the recommendation process when traditional recommendation techniques are applied. Moreover, The focus on recommending the popular items will lead to increase the undesired products, which may cause a limitation on the sales items and catalogue coverage (Jannach, Lerche, Gedikli, & Bonnin, 2013). Hence, The long tail recommendation problem makes the recommendation tool essential in delivering items that are not popular among users but which may interest a target user (Celma, 2008).

Recommending popular items does not get a lot of attention and does not provide many benefits to either users or content suppliers. In contrast, recommending items with less popularity adds serendipity and coverage to recommender systems, but it is more difficult to tackle this task (Cremonesi, Koren & Turrin, 2010). Hence, more recently, long tail recommendation is a widely debated issue in recommender system research. This is for two main reasons. Firstly, long tail products lead to increased profit. Secondly, there is difficulty in finding and identifying the items in

the tail and producing recommendations based on that. Several researches have proposed a way of tackling this challenge. For example, Shanfeng Wang, Gong, Li and Yang (2016) propose a multi-objective method that divided items into two categories, popular items and long tail items, and considered those item individually.

In terms of the profitability of long tail item recommendations, companies can make a profit from long tail items for two reasons (Yin, Cui, Li, Yao & Chen, 2012):

1. Due to the growth of the competitors being able to offer mainstream items, Economic principles push the profitability of them down. In contrast, items in the long tail can be sold at a higher profit margin.

2. Offering items in the long tail improves the shopping experience, by showing customers new items that they were not aware of, thereby offering a new order (Johnson & Ng, 2017).

2.12.1 Definition

The term ‘long tail’ has become more popular over time as way of describing the retail strategy of selling a large number of specific items in relatively small quantities, usually in addition to selling a small number of popular items in large quantities (Jeyshirii, 2014). The long tail is defined by the popularity of the item, using the frequency of its distribution. The figure below illustrates how the long tail problem is represented in a vector.

The long tail was a term introduced by Anderson (2007) to refer to instances where niche products become more popular until they eventually account for a large proportion of total sales. Hence, finding the items in the niche markets has two

important purposes: making everything available and helping users find it (Anderson, 2007). The Internet has made it easier for individuals to purchase niche products. Indeed, online retailers such as Amazon base their success on their ability to supply niche products, and it is this long tail effect that sets them apart from their competitors (Hervas-drane, 2008). However, in order for such an approach to be successful, two criteria must be satisfied: retailers must have the ability to supply a broad range of products and customers must be able to find them easily (Anderson, 2007).

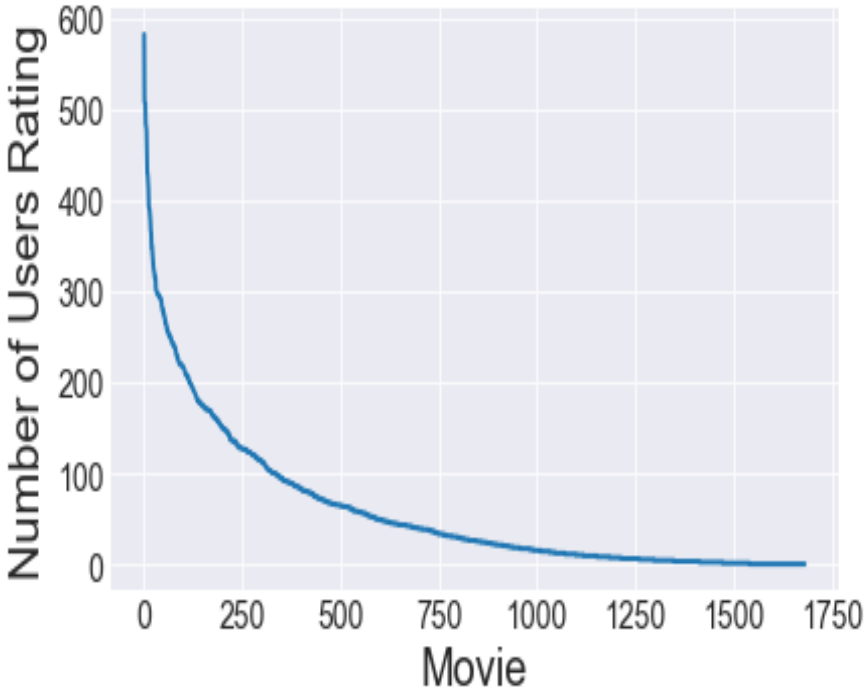

Figure 5: The long tail of rating distribution

In business terms, the long tail is the total number of non-hit items in sales (Jeyshirii, 2014). In this thesis scenario, it is the number of ratings that an item has received from users. If it is low compared with other items in the dataset, the recommender system finds it very difficult to filter and suggest this item to a

particular user. This may cause difficulty in offering some interesting items. Therefore, it affects the prediction accuracy of the recommendation.

2.12.2 Related work

Recommender systems play an important role in helping users to find and explore more items. Hence, one of the recommender system's goals is to promote the tail by suggesting a relevant and personalised recommended item. In addition, recommender systems not only promote sales growth but can also help to improve margins by providing customers with access to niche products that are likely to be less well known (Celma & Herrera, 2008). However, the growth in available items makes it very difficult to offer a prediction for those items. Hence, the main question is to establish if the recommender system algorithms can overcome this problem and discover as many items as possible.

The literature presents a number of methods for solving this problem, such as through the use of clustering techniques in order to boost item ratings in the long tail. Jeyshirii (2014) advocates using an adaptive clustering recommendation approach that groups individual items according to how popular they are. This enables items that have received only a small number of ratings to be grouped alongside similar but significantly more popular items. Cremonesi, Koren and Turrin (2010) use matrix factorisation algorithms and a neighbourhood method to evaluate the performance of the recommendation of items in the long tail. Graph-based algorithms have also been proposed (Yin et al., 2012), employing user-item information with undirected edge-weighted graphs for long tail item recommendations. Craw, Horsburgh and Massie (2015) use a CBR system and suggest that unknown artists and tracks are recommended. The system proposed in

their study could identify whether an item resides in the long tail and if it is attempting to improve its metadata with the addition of tag knowledge.

Fleder and Hosanagar (2009) studied the effect recommender systems have on sales concentration. They propose an analytical model of consumer purchases based on recommendations derived from a recommender system. The recommender system operates in accordance with a popularity rule, suggesting items that are known to sell particularly well. The result of applying this method is that sales become highly concentrated in a relatively small number of products over time. One disadvantage with this approach is that it makes no allowance for consumer preferences. Nor does it offer any scope for enhancing recommendations for items in the long tail. (Gedikli & Jannach, 2010) propose a method that is based on the rating frequencies in a way that how often this particular user rate the items and what value this item often get. It can enhance the sparsity problem and improve prediction accuracy.

2.12.3 Difficulties of long tail recommendation

The challenge involves filtering and finding the right item for the right user among a vast number of products. Popular products can be provided as a dense matrix that fits well with many recommendation techniques such as collaborative filtering, matrix factorisation, clustering and traditional algorithms. Yin et al. (2012) point out that traditional recommendation techniques work successfully with items that are familiar to a given user.

A critical issue is whether recommender systems are able to recommend items in the long tail so that users can find them, effectively meeting their interests and discovering items that they would not have discovered on their own. Such discoveries can lead to changes in sales distribution in the supplier company.

A bipartite graph, which is an adjacency matrix representation, has been proposed as a stochastic matrix to find items in the long tail region using the Markov process through a probability iteration (Johnson & Ng, 2017). The authors added genre as a new dimension to the datasets to allow for more exploration of items in the long tail and the study indicates that the extra information about items helped in reducing the problem of items in the tail.

However, a major challenge that have been found in the literature is the accuracy of the recommendation algorithms is decreased when recommending the items in the long tail (Yin et al., 2012).

2.13 Recommender system evaluation

A variety of evaluation methods have been used in the domain of recommender systems. It is only by studying recommender systems that new and improved methods can be devised (Beel & Langer, 2015). Depending on the aim of the system, an evaluation can be conducted. For example, there are two main aspects: first, the prediction of the item and how it meets the user's interests, which is called 'rating prediction'. This considers only the observed ratings and is measured by mean absolute error (MAE) and root mean squared error (RMSE). Secondly, ability to provide a list of items that are recommended to a particular user. This is called 'ranking', and is often quantified in terms of precision and recall. In each case, there are common matrices that are applied in numerous research articles. However, there are different processes of evaluation. There are three main methods, and these are explained in more detail in the next sections. The choice of method relies on the experiments that are conducted by the researchers, and whether they use real systems or public datasets.

2.13.1 Offline evaluation

Due to the availability of the dataset, this type of evaluation is the easiest to perform in experiments using an existing dataset that provides user information with ratings for some items (Adomavicius et al., 2015). The majority of the existing works on recommender system evaluation mainly focus on predictive accuracy using offline analysis (Herlocker, Konstan, Terveen & Riedl, 2004b). Such evaluation has the advantage that there is no need to ask real users to participate, which gives the developers more options to run the experiment with more than one algorithm and different datasets. It is used to conduct both prediction and top *N* recommendation. Most of the research in recommender systems focuses on offline analysis of predictive accuracy and assessing the effectiveness of recommendations. Ahn (2008) applies MAE to measure the effectiveness of a new method of solving the cold start recommendation problem. Hannon et al. (2011) use offline evaluation to recommend followers on Twitter based on the profiling of each user.

Offline evaluation provides a typical way of measuring the performance of recommendation algorithms. It also provides the ability to conduct a more cost-effective comparison between proposed recommendation methods (Cursada, Carrera & La, 2012). Shenghui Wang et al. (2014) apply MAE to an evaluation of the accuracy of semantic-enhanced content-based recommendation. More recently, Wasid and Ali (2018) evaluate multi-criteria rating against traditional CF techniques using predictive accuracy MAE.

Offline evaluation can be divided into two main recommendation evaluations: a list of top *N* recommendations and rating prediction. In each evaluation method, two factors are mainly considered: efficiency and effectiveness. In terms of

effectiveness, rating prediction is widely used, which calculates the predicted rating for the missing ratings and evaluate the algorithms that predict these values. In other words, the predictive accuracy error metric is widely applied in offline evaluation, which includes MAE, MSE and RMSE. It measures the differences between the actual rating and the predicted rating.

The efficiency aspect, on the other hand, is measured by the amount of time spent processing the algorithm or the response time in terms of generating the recommendation. However, the effectiveness evaluation plays a more important role than the efficiency evaluation for recommendation approaches.

In spite of the fact that recommendation accuracy is an important task in making an effective recommendation algorithm, user satisfaction also needs to be considered as an important factor in an evaluation. More specifically, on the occasion of developing a new system a certain interaction observation is required to measure the effect of the system. Some users might find the recommendation is very relevant and others may not find it useful.

2.13.2 Online evaluation

In an online recommender system, users are influenced by the system and directly interact with the system, and the designer aims to measure the behaviour of users while using the system. Therefore, it can reflect how well the system is suggesting relevant/interesting products (Shani & Gunawardana, 2011).

It was from online adverts that online evaluations first emerged, and these have since been used to gauge acceptance of recommendations offered by recommender systems. Click-through rates (CTR) are used as a proxy for acceptance rates. For

example, if 1,000 recommendations are made and eight are clicked, the click-through rate is 0.8%. Alternative measures of acceptance include the ratio of items bought or the ratio of downloads. Acceptance can be implied because if an individual has clicked, downloaded or bought an item that has been recommended then they presumably found that recommendation to be useful. However, this is not necessarily the case, because it is possible that a user may buy a product and later realise that it is not suitable for their needs. If the purpose of the recommender system is simply to maximise revenue, however, performance can be measured using metrics such as CTR (Beel & Langer, 2015).

More indirectly, many businesses and organisations run controlled experiments on their systems in the form of A/B testing or alternative techniques (Ron Kohavi, Longbotham, Sommerfield & Henne, 2008). Usually, these experiments redirect a fraction of the traffic of a platform towards the evaluated system and measure system performance by means of user engagement metrics such as page views, CTR (Garcin & Faltings, 2013) or, more directly, the economic benefit of the system (Shani, Brafman & Shimony, 2005). This kind of evaluation provides the strongest evidence because it is performed in real settings with real users. However, the results of these kinds of experiment must be analysed carefully in order to draw reliable conclusions and discard differences that may be caused by external factors or chance. Moreover, there is a risk involved in performing evaluations in real systems, because testing under-performing systems may adversely affect the experiences of real customers.

It is often the case that experiments such as these are costly, because devising systems for online tests is laborious. In addition, it is necessary to test the

algorithms prior to presenting the results to users as this prevents users from being subjected to negative experiences. For instance, if users are subjected to unsuitable recommendations during the test, they are unlikely to return to use the final system in future. In addition, if recommendation systems are incorporated into applications prior to deployment, there is no possibility of conducting tests. Therefore, it is imperative that tests of algorithm performance can be conducted offline whilst closely mimicking actual online behaviour (Shani & Gunawardana, 2011).

2.13.3 User study

In this type of evaluation, a user is involved in the experiment, performing a real task on the system. This type of evaluation can measure user satisfaction and get feedback about the system. Typically, users are afterward asked to express their experience through a survey. This kind of evaluation incorporates reality features, as real users examine the system.

Konstan and Riedl (2012) argue that the evaluation of a user study needs a broader set of measures to prove the effectiveness of the system. In a user study, the evaluation needs to be done using both the developed algorithm and the user interface. (Pierrakos, Paliouras, & Ioannidis, 2012) provides a user evolution based on measuring user interaction with a system using both qualitative and quantitative methods. Armentano, Christensen and Schiaffino (2015) apply the technology acceptance model to measure user satisfaction and evaluate user acceptance of the recommender system in a movie domain.

Despite the fact that a user study evaluation delivers a wide range of user expectations, goals and objectives relevant to designing a framework for recommender systems, it is a challenge to compare the new system with a

benchmark or an existing one and obtain reliable results. Objectives differ from user to user; for example, some users may consider trust in the system important, while others need a variety of recommendations that are tailored to their preferences. Hence, it is a challenge for researchers to handle the user experience when developing a new system (Konstan & Riedl, 2012).

2.14 Summary

This chapter has reviewed the existing studies related to the problems of sparsity and long tail recommendation. It has also discussed and formalised the recommender system problem. Recommender systems consist of two main components: users and items. Users provide their opinion about the items. The users' profiles are defined based on their opinions about items, e.g. ratings and information (e.g. demographic information); whereas items' profiles are defined based on the item's features, e.g. genre. By investigating more about users' and items' profiles, a recommendation algorithm can solve the problem of recommending items with insufficient ratings to users. A classification of the existing recommendation algorithms based on their principal characteristics was then provided. The main benefits as well as the pitfalls of the recommendation algorithms were discussed by highlighting several factors under which their performance would suffer. At the end, the limitations of the state-of-the-art algorithms and the algorithms that have been used to benchmark the proposed method were discussed.

It can be seen that previous research suggests a number of approaches in collaborative filtering, content-based filtering and hybrid recommendation. However, most of the existing recommendation algorithms do not apply case-based

reasoning methods to solve the long tail problem in recommendation. In addition, the multi-level method has also not been integrated with the switching method to further improve prediction error and solve the long tail problem.

The next chapter presents the proposed recommender system framework based on a switching hybrid method that adopts a multi-level algorithm in order to solve the long tail recommendation problem.

3 Research Methodology

3.1 Introduction

This chapter explains the followed methodology that was used in the experiments, involving collaborative filtering and content-based filtering techniques. The proposed system framework is described in detail along with its formalism. The datasets that were used in this thesis are explained. Finally, the chapter provides a detailed comparison of the baseline algorithms and the proposed method.

The main aim is to explore the specific challenges that recommender systems still suffer from and to discover how changing similarity measures can further improve the recommendation process. This thesis proposes a novel method of solving the long tail recommendation problem, an alternative way of dealing with items with insufficient ratings. insufficiency of data leads to the long tail recommendation problem which affects producing a good predictive model in the tail. Therefore, when the system has failed to obtain the rating using ratings-based CF, it switches to a content-based method utilising user history to obtain similarity scores according to the item attributes in the user's profile. This is described in more detail in the next section.

3.2 Switching hybrid multi-level recommender system framework

The proposed method enables the simultaneous use of two techniques in recommender systems, thereby eliminating weaknesses associated with each technique. In this way, it aims to increase the accuracy of predictions made by the recommender system. A hybrid methodology is developed with the implementation

of CF, CBF and CBR. In this hybrid, a solution is presented approach to the long tail problem encountered in the CF method.

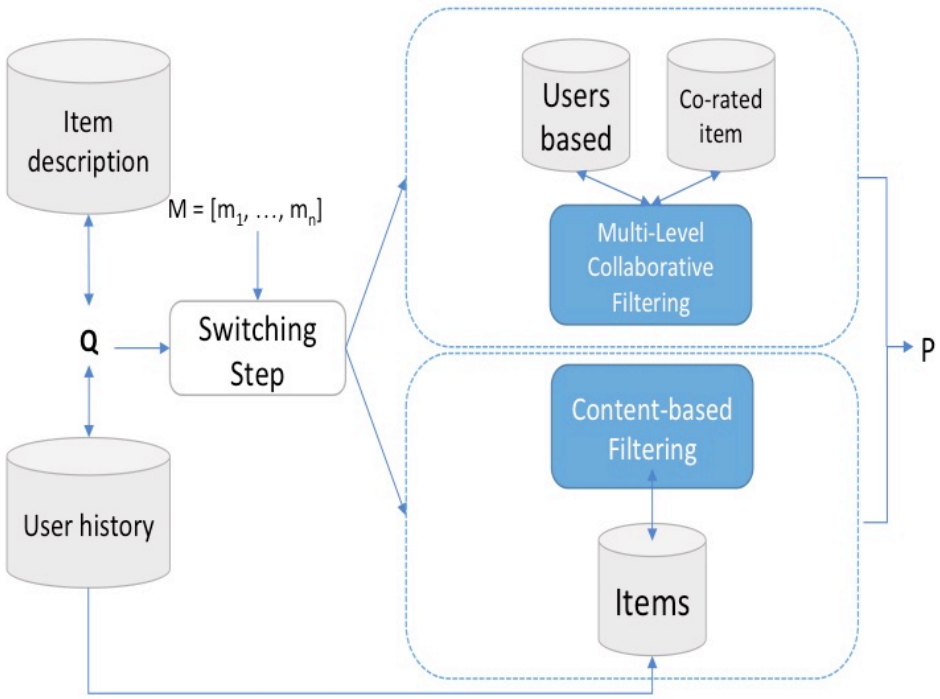

Figure 6: A switching hybrid multi-level recommender system framework

This section explains the hybrid system that was designed using the Java programming language. As can be seen in Figure 5, the architecture of the methodology has two main modules: a CF component that calculates the predicted rating based on other similar users and CBF. In the content-based component, the rating prediction is calculated using other similar movies that the user has rated in the past. Both modules were implemented using the jCOLIBRI framework that is an open Java source for building a CBR system. It provides a prototype with CBR components and interfaces to implement (Recio-García & González-Calero, 2014).

The system receives a query (Q) that identifies the target user (u) and movie (m). Ratings are on a scale from 1 to 5 and the aim is to compute the estimated rating for the movie r(m,u)′.

$$Q =< u, m > \quad (2)$$

The first step in this system is to decide which methods are most effective at correctly calculating the rating prediction. This decision is based on the number of ratings received by the target movie. In order to make this decision, the system computes a vector (R_m) that represents the rating of a concrete movie m.

$$R_m =< m, r >_m = (< m, r(m, u_1) >, \ldots\ldots, < m, r(m, u_j) >) \quad (3)$$

In the first step, the system obtains the number $|R_m|$ of ratings that the query movie (m) has. It then compares this value with a threshold constant (δ). If the number of ratings for m is higher than δ, then this movie is not in the long tail problem and a collaborative filtering module can be used. On the other hand, if the number of ratings is lower than δ, the system cannot find similar users that rate this movie. This is caused by a lack of ratings having been provided for a particular item. Therefore, the system will switch to using the content-based module based on the user's search history.

The experiment was repeated with a content-based method based on users. This method uses the statistical average of ratings for each genre defined in each user description.

The next section explains each module which has been used as a baseline algorithms in more detail and sets out how the rating prediction of a movie is calculated.

3.2.1 CF module based on users

The first module used in the hybrid system is a collaborative filtering CBR system. The main goal of this module is to calculate the rating prediction based on the ratings of other similar users u′. This module compiles a list of all of the movies previously rated by any user (R_u) in order to obtain the user similarity.

$$\boldsymbol{Q_{CF}} = < \boldsymbol{u}, \boldsymbol{m}, \boldsymbol{R_u} > \tag{4}$$

where

$$R_u = < m, r >_u = (< m_1, r(m_1, u) >, \ldots\ldots, < m_n, r(m_n, u) >) \tag{5}$$

$$r \in [1..5]$$

This module uses the k-NN algorithm to calculate the rating prediction. In this case, the users obtained by the k-NN must contain a rating for the target movie *m*. To calculate the similarity between two users, the collaborative filtering module compares both lists of ratings:

$$sim_{CF}(Q_{CF}, C_{CF}) = sim(R_u, R_{u'}) \tag{6}$$

where

$$\boldsymbol{C_{CF}} = < \boldsymbol{u'}, \boldsymbol{m}, \boldsymbol{R'_u} > \tag{7}$$

This module can be configured using any similarity function that calculates the similarity between both vectors. This experiment was configured with the two most popular similarity functions: the Euclidean distance and the Pearson correlation.

When the system has retrieved the *k* most similar users that have rated the target movie *m*, it calculates the rating prediction using other rates derived from these users. This prediction is calculated with the weighted average of the rating and similarity measure.

$$\boldsymbol{r(m,u)' = \frac{\sum_{i=0}^{k} r_i(m,u') * sim_i(R_u, R_{u'})}{\sum_{i=0}^{k} sim_i(R_u, R_{u'})}} \quad (8)$$

$r(m,u)'$ is the result returned by this module as the rating prediction. The following section explains the second method.

3.2.2 Content-based module based on user history

The second CBR module is used in order to resolve the long tail problem, as described in detail in the literature review. For example, in this dataset, where there is an insufficient number of ratings from other users for the target movie *m*, this module calculates the rating using a content-based similarity function, which is based on the description of the movie rated by the user. This system creates a personal case base for the target user where each case (C_{CB}) contains a list of genres that describe the movie.

$$C_{CB} = < u, m, G_m > \quad (9)$$

$$G_m = \{g_1, \ldots\ldots, g_n\} \quad (10)$$

After being given a query, movies are compared according to the number of common genres.

$$\mathrm{sim}(\mathrm{m}, \mathrm{m}') = \frac{\mathrm{G_m} \cap \mathrm{G'_m}}{\mathrm{G_m} \cup \mathrm{G'_m}} \tag{11}$$

Using the *k* most similar movies, the CBR module calculates the rating prediction using equation 8 that is applied in the first module.

3.2.3 Combined module

In this module both recommender system methods were combined with a similarity function. In this case, the similarity between two users is calculated by the average of the similarity calculated using the collaborative filtering and content-based method.

3.3 Rating prediction

There is an existing extensive study based on the rating prediction that is using a numerical value that represents the user interest and preferences for a particular item. However, there is still some limitation on the traditional recommendation techniques such as CF and CBF. Neighbourhood recommendation is one of the most popular techniques that has been used to calculate the predicted rating for a given item based on the past ratings of similar users (Nava Tintarev & Masthoff, 2011). The system produces predicted ratings, the accuracy of which is subsequently tested (Gunawardana & Shani, 2009). The rating prediction problem arises when interactions between users and items are encoded using a rating matrix

that only has access to partial information regarding the tastes of users. In such scenarios, recommender systems offer predictions for ratings in the test subset according to the available ratings that have already been offered. Predictions are usually based on error metrics such as the root mean square error (RMSE) or the mean absolute error (MAE).

The RMSE is the most popular predictive accuracy measure in recommender systems so far. It is often related to the purpose of rating prediction; i.e. estimating the rating value that a user would assign to an item that he/she has not seen yet (Steck, 2013).

Up until recently, rating prediction was the standard approach when devising recommender algorithms. The availability of countless datasets such as MovieLens helps to explain the success of this approach. Next, is explained the most popular recommendation algorithm that is associated with the rating prediction.

3.4 Recommendation algorithms

3.4.1 k-Nearest Neighbour (k-NN)

In the CF recommendation technique, k-NN is still the most popular recommendation algorithm due to its simplicity, efficiency and ability to suggest more accurate and personalised recommendations (Desrosiers & Karypis, 2011). Neighbourhood formation is used to find the most similar users by either users to users or users to items. It is widely applied with recommender system techniques to group users that have similarity (Desrosiers & Karypis, 2011). In this research the algorithm is given the rating by similar users for many items to calculate the predicted rating. The neighbourhood similarity measures calculate the top k

neighbour users for the target user. The two most common similarity measures are Euclidean distance and Pearson.

The optimal number of neighbours was calculated through changing the neighbourhood size, starting from 3 up to 100, and the MAE was observed using all validation sets.

Rating data are required to form neighbourhoods in the CF methodology. Bell and Koren (2007) advocate the use of neighbourhood-based methods in order to make k-NN approaches more accurate without significantly extending the running time. However, in the absence of sufficient user rating data, neighbour formation and recommendations will be adversely affected. Similarities between users or items can be inferred by users' tags and topic interests (Weng, Xu, Li & Nayak, 2007). Rather than relying solely on rating data, Weng et al. (2007) used taxonomy information relating to items along with the user's previous ratings for the purposes of neighbourhood formation.

The nearest neighbour algorithm simply stores all of the training data, in this case textual descriptions of implicitly or explicitly labelled items, in its memory. In order to classify a new, unlabelled item, the algorithm compares it to all of the stored items using a similarity function and determines the 'nearest neighbour' or the k-Nearest Neighbour. The class label or numeric score for a previously unseen item can then be derived from the class labels of the nearest neighbours.

The similarity function used by the nearest neighbour algorithm depends on the type of data. For structured data, a Euclidean distance metric is often used. When using the vector space model, the cosine similarity measure is often used. In the Euclidean distance function, a feature that has a small value in the two examples is

treated the same as features that have a large value in both examples. In contrast, the cosine similarity function will not have a large value if the corresponding features of two examples have small values. As a consequence, it is appropriate for text when two documents are wanted to be similar when they are about the same topic but not when they are about unrelated topics.

The distribution of item ratings affects the market in the real-world scenario. Only a small number of items are rated frequently, referred to as popular items. The majority of the items are rated very rarely, referred to as long tail items. As can be seen in Figure 5, a number of ratings have been awarded by users to each movie. It is clear many items have a few ratings among all the items in the dataset. It is evident that most of the items have only a few ratings. This rating distribution provides an important aspect to apply the recommendation for some reasons:

1. Popular items tend to be relatively competitive with less popular items for the company. However, the less popular items produce more profit, which means that companies gain advantages from recommending less popular items. For example, Amazon makes a high profit from selling items belonging to the long tail (Aggarwal, 2016).
2. The popularity of the items may lead to a negative impact on recommendations, since users get bored receiving the same group of recommendations. Many recommendation algorithms have difficulty in observing the items in the long tail.
3. In the long tail distribution, items with a smaller number of rating have higher percentages than the most frequently rated items. This is very effective for the neighbourhood algorithms since it works on the idea

that the rating pattern between two items represents the closest item to be recommended. Therefore, the prediction results may be misleading, which affects the accuracy of the recommendation (Aggarwal, 2016).

On the other hand, the k-NN method faces a big challenge in regards to recommendations because of the sparsity issue. In reality, users often rate only a small number of items amongst a huge amount of available data. Two users may rate a small number of items, for example, in which finding a pattern is quite complex. The graph-based model and dimensionality reduction can address this issue.

3.5 Similarity measures

The key problem with similarity-based algorithms is defining similarity between users or items. This is calculated by considering the available ratings (Lü et al., 2012b). In this research the most common similarity measures are considered that have been applied in the empirical literature: Pearson similarity and Euclidean distance.

In neighbourhood-based models, similarity between users is calculated based on their historical ratings of similar items, which represent user-item associations. This does not require extensive data collection, unlike latent methods. It can be seen as a prediction of estimated value. It is achieved by finding the similarity measure between different users based on the items they have previously rated. Based on similarity scores, users with the highest scores are like-minded with users who form the neighbourhood. The k-NN is employed for the same. The items that are rated by the neighbourhood and not by users are recommended to the users. The merits of

using such an algorithm are that it is intuitive, requires no training and the relationship can easily be explained.

3.5.1 Cosine similarity

Cosine similarity utilises the angle between two vectors (users or items) to calculate the similarity. However, the author in (Shen, Liu, & Zhang, 2017) discussed the cosine similarity that utilise only the value of the user ratings and not taking into account the average of these rating, which results in producing inaccurate similarity. Hence, an adjusted cosine similarity was used to consider the average. However, the adjusted cosine similarity works as exactly the same as Pearson Correlation Coefficient (PCC) that has been used in this thesis.

In addition, the author in (S. Sun et al., 2017) reviewed some similarity measures and argued that a similarity measures affect the prediction accuracy. Furthermore, the authors (Tan & He, 2017) compared different similarity measures and presented the results in a table showing that PCC works very well in predicting the ratings using the datasets that have been utilised in this thesis.

3.5.2 Pearson correlation coefficient (PCC)

Pearson correlation similarity is employed in order to anticipate the extent to which two vectors are linearly related. $\overline{\text{Ru}}$ and $\overline{\text{Ru}'}$ are the mean values for the Ru and Ru′ vectors respectively. It is calculated using the following formula:

$$sim_{pea}(R_u, R_{\grave{u}}) = \frac{\sum_{m=0}^{|M|}(r(m,u)-\overline{R_u})(r(m,u)-\overline{R_{\grave{u}}})}{\sigma_u \sigma_{\grave{u}}} \quad (12)$$

where

$$M = R_u \cap R_{\grave{u}} \tag{13}$$

3.5.3 Euclidean distance

Based on the empirical literature, the Euclidean distance similarity metric has the best performance in the same dataset (Laveti et al., 2016). It is a measure of how close two users are if their ratings are plotted on a set of axes (Redpath, 2010). In the propsed method, it measures the distance between the query R_u and the case $R_{\grave{u}}$.

$$\mathrm{sim_{Euc}}(R_u, R_{\grave{u}}) = 1 - \sqrt{\sum_{m=0}^{|M|}(r(m,u) - r(m,\grave{u}))^2} \tag{14}$$

where

$$M = R_u \cap R_{\grave{u}} \tag{15}$$

3.6 Baseline user-based CF

CF is a widely used baseline model for recommender systems and is based on user-user relationships. Similarity profiles are used to gauge the similarity between two users and the target rating is estimated using the average rating of the closest neighbour's ratings for the same item. PCC and Euclidean distance are employed so as to measure the similarity between user profiles.

3.7 Summary

This chapter has presented the methodology of this thesis. It introduced the framework of the switching hybrid multi-level method. The switching method uses

the two most popular recommendation methods, collaborative filtering and content-based filtering that has been used as a baseline algorithm in the thesis. This include a way of how the switching method is implemented using a predefined constraint. A user profile is used when there is no sufficient rating for the target item, which means the item belongs to the long tail. In this case, user-based CF has failed to calculate the predicted value. The chapter then presented the details of the methods that are used in this thesis as a baseline for comparison with the proposed method. The algorithms that were applied in this thesis, along with the similarity measures, were then explained in more detail. Finally, the similarity measures that have been used in the experiments are presented.

4 Experimental evaluation

This chapter represents the experiments realized to evaluate the proposed method with the details of all the steps that were used in the experimnts. In order to improve the recommendation algorithms and measure the developed algorithms against the baseline, meaningful experiments producing valid comparisons using the right metrics are required. In addition, an appropriate dataset is needed to deliver meaningful recommendation methods. This chapter explains the experiments that were conducted to evaluate the chosen methodology with the CBR approach and multi-level method. It first details all of the steps in the experiment with the dataset that was used in each experiment. The evaluation of the proposed switching method using the CBR and multi-level method is then presented. This is followed by an explanation of the multi-level method using the triangle similarity measures. Finally, the feature combination method is explained, which utilises demographic filtering and collaborative filtering using four classifiers: k-NN, Random Forest, Neural Network and AdaBoost. All the evaluation in this thesis focuses on predictive accuracy measures, since previous studies widely apply MAE and RMSE to evaluate their algorithms using offline evaluation. This thesis presents a clear comparison with the baseline method that proves the robustness of the results compared with traditional prediction calculations. The use of k-NN is also justified as an appropriate algorithm in this thesis. The results and findings are then discussed in more detail.

4.1 A switching CBR experiment

In this experiment, the CBR approach was used to switch between the appropriate methods according to certain criteria which is refer to ICCBR17. Items with few

ratings were considered a major problem, reducing the prediction accuracy and affecting the overall recommendation. A 'user history' was a solution to this, which involves finding a close item that can help find the estimated value for the prediction. The proposed method alleviates the popularity bias and finds a way of solving the problem of items that belong to the long tail that collaborative filtering suffers from. For example, it does not predict correctly the items that contain a small number of ratings over the large item populations with respectively large number of ratings. This method presents the user history as a case base can help to reduce the long tail recommendation problem as well as increase the probability of identifying similar items for user's past ratings of similar items. These cases can be identified using case-based reasoning on top of a switching hybrid recommender system. This method was experimented with a freely-avilable dataset of more than 100,000 cases, which is explained in the next section.

4.1.1 MovieLens 100K Dataset

Selecting the datasets is an important step that leads to the successful evaluation of a recommender system algorithm (Herlocker et al., 2004b). This research uses the MovieLens dataset (Harper & Konstan, 2015) which is a commonly used dataset in the domain of recommender systems. It contains the results of real users' interactions with the MovieLens recommender system website. It has the ability to recommend movies using the user profile. The availability of the content descriptions helps in finding similar movies to the one selected. It is designed to offer user-item matrices to be used to develop recommendation algorithms. This dataset was chosen to evaluate the developed algorithms in this thesis, and show their effectiveness and the novelty compared with the benchmark algorithms. The

dataset consists of 100,000 ratings using numerical value ranging from 1 to 5, from 943 users based on 1,682 movies (Aggarwal, 2016). Users who had made fewer than 20 ratings were removed from the analysis. In addition, this dataset contains demographic information for each user including details of their age, gender and occupation. Users who did not complete these details were removed from the analysis. It was collected via the MovieLens website (movielens.umn.edu). The following table provides details for the rating file:

UserID	MovieID	Rating	Timestamp
1	1193	5	978300760
2	1193	5	978298413
12	1193	4	978220179

Table 4: MovieLens dataset information

This dataset has been widely used in the empirical literature for the purposes of improving prediction accuracy (Laveti et al., 2016; Qian, Feng, Zhao & Mei, 2014).

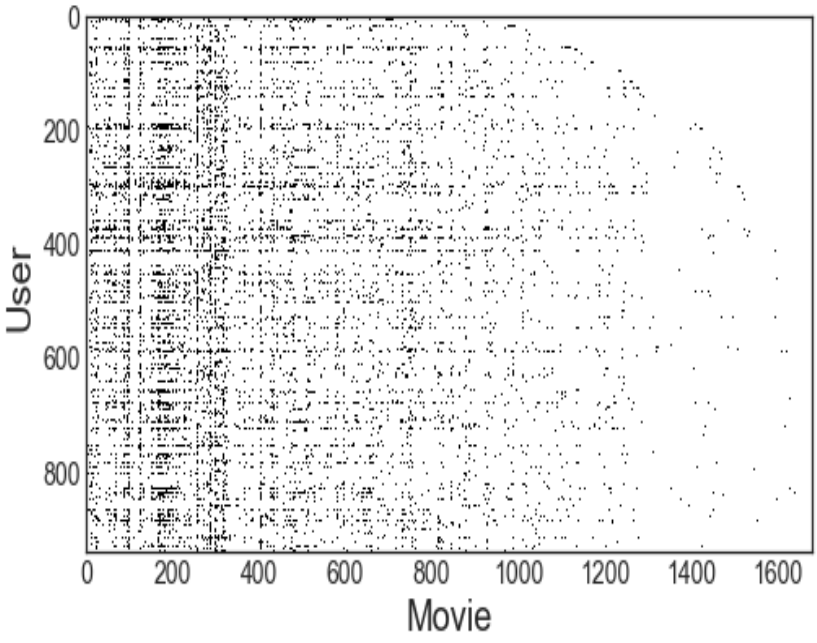

Figure 7: User x movies sparsity

As can be seen in Figure 7, the sparsity visualisation is very clear. It was created by a matrix with dimensions $n_{users} \times n_{movies}$ as presented in Equation (16), where each element r_{ij} represents a single rating by user a of movie i. This matrix indicates how users have rated the available movies. The matrix shows that the dataset is sparse and most users have rated only a few movies. Therefore, items with few ratings need to be considered and the sparsity issue require a solution in recommender systems. The new items that have been added recently to the system may also cause this problem.

The matrix density is calculated as follows:

$$\text{density} = \frac{\mathrm{n_{ratings}}}{\mathrm{n_{users}} \times \mathrm{n_{movies}}} \tag{16}$$

In this dataset the average density is 0.063.

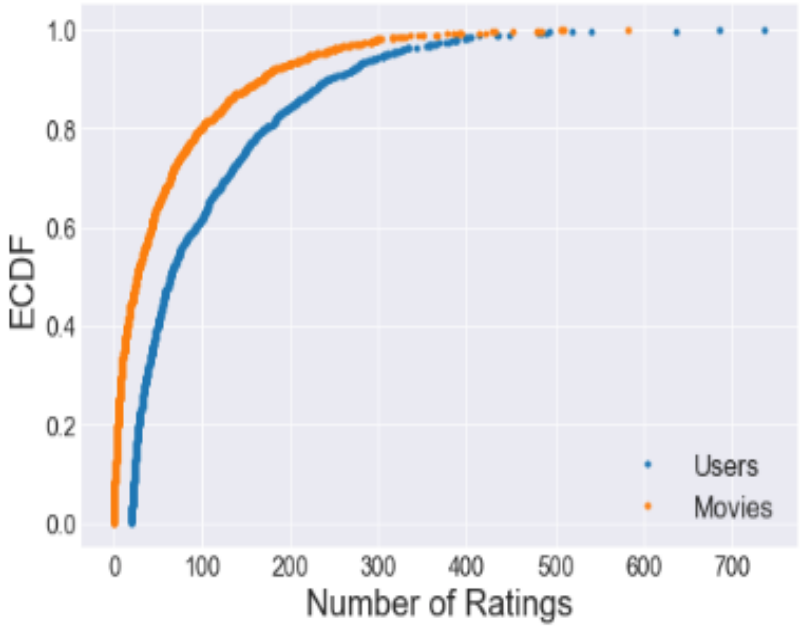

Figure 8: Distributions of ratings by user and movie

In Figure 8, an Empirical Cumulative Distribution Function (ECDF) was conducted to visualise the distribution of number of ratings by users and movies. ECDF was chosen as it provides a very meaningful representation of the long tailed distribution compared with a histogram.

It can be seen that almost 40% of all users rated 50 or fewer movies. 90% of movies have 160 or fewer ratings. Overall, it can be seen that a large fraction of movies and users have few ratings associated with them; by contrast, a small number of movies and users have many more ratings.

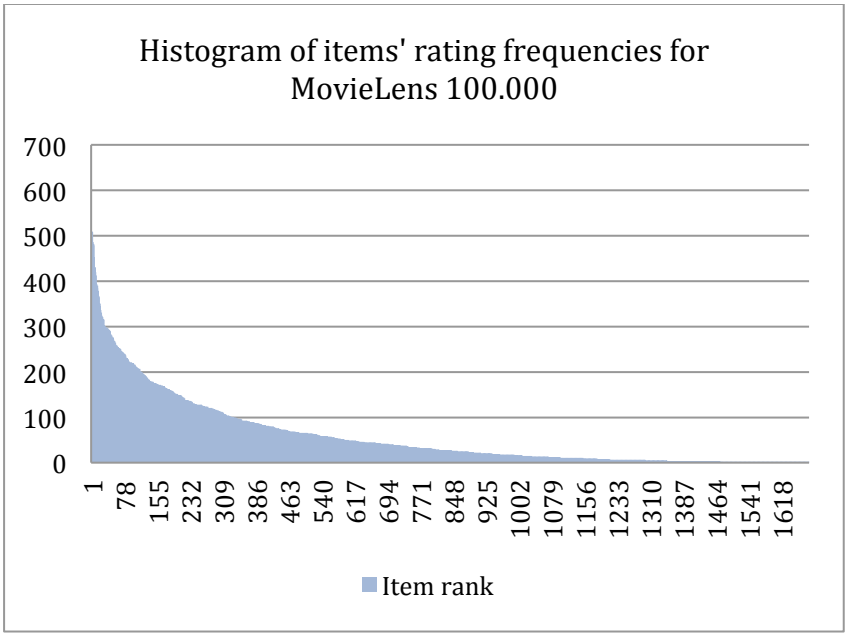

Figure 9 Histogram distribution of item's rating frequency using 100.000 MovieLens dataset

Figure 9 presents the long tail distribution using the item's ratings popularity and shows the items that belong to the long tail for the 100.000 movies rating. Hence, the long tail distribution is computed by the total number of ratings recived for each item. The x-axies contains the list of movies ranked by its total ratings. For example, movie1 has more than 400 ratings. In this senarion, items can be divided

into two set the head and the tail by determining the cutting point along the x-axis of the distribution. For example, in this thesis the cutting point is selected to be 10 which divided the items into two sets. The first on is the items that have been rated many times and become popular among users. The second items that have few ratings, which are belonging to the long tail.

The main point to be considered from the analysis above is that the data is quite sparse regarding the ratings; a new model needs to be presented that offers a possible solution to the items in the long tail or that have few ratings.

4.1.2 Experiment setup

This experiment applies a leave-one-out method, which is a method obtained by setting $k = 1$ in the leave-k-out method. For each individual active user, one rated item is withheld in turn. The learning algorithm operates using the remaining data and the withheld information is used for test purposes in order to gauge the accuracy of subsequent predictions. However, such an approach is prone to overfitting and the degree of computational complexity is unusually high. Such an approach is appropriate for appraising the quality of different models when users have already registered as members of the system (Cremonesi, Turrin, Lentini & Matteucci, 2010).

For each query the differences between the rating predicted by the system and the actual rating was calculated. This evaluation was executed with the following condition

All tests were varied by systematically by changing two elements: the first being the similarity function used; the second is the k in the k-NN algorithm. The similarity

functions used in the experiment are PCC and Euclidean distance, explained in section 3.5. The value of *k* is: 3, 5, 10, 15 and 25.

4.2 Switching multi-level method

In this experiment, the multi-level method was integrated with the CF approach using a switching method referred to as MLCBR. The number of co-rated items reflects the degree of connection between users. For example, a high number of co-rated items might indicate a high level of similarity between two users. Traditional similarity metrics do not consider the number of co-rated items. Hence, the experiment adopted the multi-level method to more precisely identify the nearest users through a pre-defined level. In addition, a switch method was successfully applied to enhance the similarity value of items that belong to the long tail. This enhances the process of k-NN by finding a large margin within an application.

$$Sim_{\mathrm{a,b}}^{Proposed} = \begin{cases} Sim_{\mathrm{a,b}}^{PCC} + x1, if \ \left|\frac{Ia \cap Ib}{T}\right| \geq t1 \ and \ Sim_{\mathrm{a,b}}^{PCC} \geq y \\ Sim_{\mathrm{a,b}}^{PCC} + x2, if \ \left|\frac{Ia \cap Ib}{T}\right| < t1 \ and \ \left|\frac{Ia \cap Ib}{T}\right| \geq t2 \ and \ Sim_{\mathrm{a,b}}^{PCC} \geq y \\ Sim_{\mathrm{a,b}}^{PCC} + x3, if \ \left|\frac{Ia \cap Ib}{T}\right| < t2 \ and \ \left|\frac{Ia \cap Ib}{T}\right| \geq t3 \ and \ Sim_{\mathrm{a,b}}^{PCC} \geq y \\ Sim_{\mathrm{a,b}}^{PCC} + x4, if \ \left|\frac{Ia \cap Ib}{T}\right| < t3 \ and \ \left|\frac{Ia \cap Ib}{T}\right| \geq t4 \ and \ Sim_{\mathrm{a,b}}^{PCC} \geq y \\ Sim_{\mathrm{a,b}}^{PCC} \quad , if \ \left|\frac{Ia \cap Ib}{T}\right| < t4 \ otherwise \end{cases} \quad (17)$$

The experiment in this method was conducted using the algorithms that are detailed in section 3.2 to make a comparison with the baseline CF and CBF as a benchmark to evaluate the proposed method. However, the following algorithm was also added to the comparison:

4.2.1 Multi-level

This algorithm relies on PCC similarity measures. It is based on multiple levels, from the top to the bottom, with each of these levels having a number of constraints that are defined in the equation, where $sim_{a,b}$ denotes the similarity between user a and user b. T stands for the total number of co-rated items; $t1, t2, t3$ and $t4$ are the threshold of co-rated items for user similarity. It is considered that $t1 = 50, t2 = 20, t3 = 10$ and $t4 = 5$. In addition, x is defined as $x1 = 0.5, x2 = 0.375, x3 = 0.25, x4 = 0.125$ and $y = 0.33$.

4.3 Triangle similarity using the multi-level method for item-based CF

The aim of item recommendations is to find the items that are most likely to be relevant to a particular user. In order to create the recommendation for a particular user, the recommendation algorithm known as k-NN finds the nearest similar items that have the most similarity measures to the target item. To determine an item neighbourhood for an active user, the similarities between two items must be measured. This distance or similarity can be estimated using various kinds of proximity measures such as cosine similarity and PCC. In many cases, ratings are the key issues that are used to calculate similarity and predict unknown items.

Item-based collaborative filtering was developed using a triangle similarity measure. The main aim of this experiment was to improve the prediction accuracy compared with the traditional measures. Item-based collaborative filtering looks into a set of items that the target users have rated in the past and computes how similar they are to the target item; it then selects the k most similar items. Once the

most similar items are found, the prediction is computed using a weighted average of the target users' ratings. Hence, there are two main aspects to be considered: similarity computation and prediction generation.

As the number of co-rated items reflects the degree of connection between users, a high number of co-rated items might indicate a high level of similarity. In contrast, traditional similarity metrics do not consider the number of co-rated items (Shen, Liu & Zhang, 2017). Hence, there are some limitations in the traditional similarity metrics in terms of data sparsity and coverage, which affect the quality of the recommendations. To solve this problem, triangle similarity measures are proposed by S. Sun et al. (2017); these result show an improvement in accuracy when combined with co-rating. Triangle similarity is integrated with certain constraints that apply a number of co-rated items. It takes into account the length and the angle of rating vectors between users and allows for positive and negative adjustment using a multi-level method. Therefore, in this thesis, a hybrid method was proposed that adopts a multi-level CF approach, enhancing the similarity value of users that belong to certain categories and ignoring the rest (Polatidis & Georgiadis, 2016). It enhances the process of k-NN by finding a large margin within an application. The triangle similarity measure is defined as follows:

$$Sim_{a,b}^{Tri} = 1 - \frac{\sqrt{\sum_{u \in c_{a,b}} (r_a - r_b)^2}}{\sqrt{\sum_{u \in c_{a,b}} r_a{}^2} + \sqrt{\sum_{u \in c_{a,b}} r_b{}^2}} \quad (18)$$

The value range is between 0 and 1, where the closer a value is to 1, the more similar it is. The triangle approach considers both the length of the vectors and the angle between them, so it is more reasonable than angle-based cosine similarity.

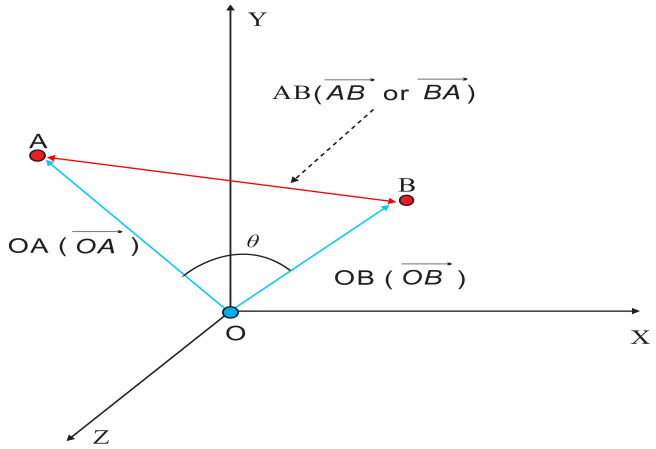

Figure 10: A triangle in three-dimensional (3D) space (S. Sun et al., 2017)

For example, if the two vectors A = 5,5,5 and B = 1,1,1 are given, then cosine similarity is 1. In contrast, the triangle similarity between them is 0.33.

$$Sim_{a,b}^{Proposed}$$

$$= \begin{cases} Sim_{a,b}^{Tri} + x1, if \left|\frac{Ia \cap Ib}{T}\right| \geq t1 \; and \; Sim_{a,b}^{Tri} \geq y \\ Sim_{a,b}^{Tri} + x2, if \left|\frac{Ia \cap Ib}{T}\right| < t1 \; and \left|\frac{Ia \cap Ib}{T}\right| \geq t2 \; and \; Sim_{a,b}^{Tri} \geq y \\ Sim_{a,b}^{Tri} + x3, if \left|\frac{Ia \cap Ib}{T}\right| < t2 \; and \left|\frac{Ia \cap Ib}{T}\right| \geq t3 \; and Sim_{a,b}^{Tri} \geq y \\ Sim_{a,b}^{Tri} + x4, if \left|\frac{Ia \cap Ib}{T}\right| < t3 \; and \left|\frac{Ia \cap Ib}{T}\right| \geq t4 \; and \; Sim_{a,b}^{Tri} \geq y \\ 0 \qquad , if \left|\frac{Ia \cap Ib}{T}\right| < t4 \; otherwise \end{cases} \quad (19)$$

Compare the similarity given by $Sim_{a,b}^{Tri}$ using y, which is predefined to be equal to 0.33 and the number of co-rated items. If it is less than a specified level t, then go to the next level and continue going to the next level until such time as the right level is found. Otherwise, assign the similarity to be 0.

The similarity measure is important for finding accurate items in recommender systems, and more specifically for calculating the similarity between two users or

items and finding the closest one. It is a big challenge to determine the distance measures that accurately find similar users. It significantly relies on the rating aspect, which allows users to assign a high or low rating to a certain item based on their preferences and behaviour. Basically, to compute the similarity between items, the first step is to determine users who have rated both items and who have the most similar items with similar ratings. Many different measures have been used and modified to improve the accuracy of this process. However, this is the first method that employs the triangle measure with multi-level algorithms, and it is proven that it outperforms the Pearson correlation measure.

The results for the three datasets with different parameters are given below. All algorithms were implemented in the Java programming language. In this experiment, *k* represents the number of nearest neighbours.

4.4 Real dataset

The experiment was run with three real datasets in order to compare the results with different parameters, such as the number of items and users. All datasets were evaluated using cross-validation with five folds and *k* is the number of neighbours, specified to be equal to 3, 10, 30, 50 and 100.

4.4.1 MovieLens 100K

This dataset is explained in section 4.1.1.

4.4.2 MovieLens 1M

This dataset contains 1,000,209 anonymous ratings of approximately 3,900 movies made by 6040 users. The University of Minnesota created an online movie recommendation system, and its items are rated by users who joined MovieLens in

2000. All ratings are on a scale between 1 and 5. This dataset is also publicly available for running offline experiments and is widely used for collaborative filtering recommender systems (Harper & Konstan, 2015).

4.4.3 Yahoo! Movies

This is a dataset obtained from Yahoo Labs under license. It contains 7,642 users, 11,915 movies and 211,111 ratings. The rating scale is between 1 and 5.

4.5 Offline validation

This experiment was developed and run using an open Java source called 'recommender101'. It is an easy to use framework that allows for the implementation of recommender system algorithms and metrics to carry out the offline evaluation of the recommendation methods. Most known algorithms are implemented, such as k-NN and matrix factorisation. All the settings and parameters are defined in a configuration file, which allows the user to apply different settings and run the experiment several times and compare the results. The framework is easily extendible and gives an opportunity for the user to implement a new recommender method and new metrics. Moreover, it provides cross-validation, in which a number of folds can be changed, and includes evaluation metrics such as precision, recall, MAE, RMSE and so on (Jannach, Lerche, Gedikli & Bonnin, 2013).

Since the historical user data already exists, the machine learning algorithms could be conducted, along with the standard evaluation metrics used to validate the algorithm and compare it with the new one. This experiment was conducted using three real datasets in order to compare the results with different parameters, such as

the number of items and users. All datasets were evaluated using cross-validation with five folds and *k* is the number of neighbours, specified to be equal to 3, 10, 30, 50 and 100.

4.6 Demographic attributes using k-NN, Random Forest, neural Network and AdaBoost

This section, proposed a method that approves the differences on combining the demographical features using four classification algorithms; k-NN, Random Forest, Neural Network and Adaboost. The main idea of this experiment, which is not found in other works in the literature, is to have a feature combination method that utilises a hybrid recommendation approach. In addition, this method presented the evaluation of different classifiers in order to identify which classifier performs better when demographic data are integrated into the recommendation process.

As can be seen in the literature, the sparsity issue is a major challenge for recommender systems in terms of producing the right recommendation for the right users. This issue has been further expanded due to the growth of items available and of users with few ratings and little user information. This leads to difficulty in finding similarity between two users. In this experiment, a feature combination hybrid approach is proposed to solve the sparsity problem through reducing the error rate, using four classifiers and comparing the results. The combination was obtained through matching user demographic attributes with the user rating CF method, as shown in Figure 11.

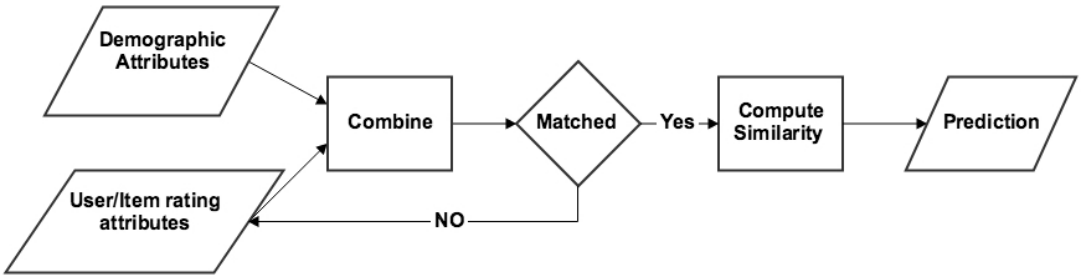

Figure 11: Feature combination architecture

In order to evaluate the results, the experiment is conducted on the MovieLens dataset using the available demographical attributes, age, gender and occupation. Those attributes are defined as categorical and represented for each user in the rating vector. This can help in finding similar users to improve prediction accuracy. The profile vector is represented at the attribute level in order to compute the similarity.

The similarity is then calculated between the active user and the closest one. Next is the final step of calculating the predicted rating. This experiment was evaluated using cross-validation with number of folds = 10. In summary, the steps of the proposed method are:

CF is combined with demographic attributes (age, gender and occupation) to find similar users. The combination was made through matching user ID from user-item rating data with user demographic data as detailed in the algorithm below, where row [0] and line [0] represent user ID, row [1,2,……,N] represents the attributes in CF, and line [1,2,…..,N] represents the attributes in DF.

After matching each user with the demographic attributes, the similarity is computed using four classifiers, k-NN, Random Forest, Neural Network and Adaboost.

The final step involves the calculation of the predicted rating, which is then compared with the actual rating, and the difference is calculated.

Algorithm 1 CF+DF

```
1: Input: user-item rating attributes file (f1). demographic attributes (f2).
2: Output: user demographic attributes with item rating (f3).
3: for <row in f2> do
4:     for <line in f1> do
5:         if row [0] == line [0] then
6:             f3 = row [0,1,..,N] + line [1,2,...,N]
7:         end if
8:     end for
9: end for
```

The predictive accuracy is applied to evaluate the proposed experiment. Both mean absolute error (MAE) and root mean squared error (RMSE) are used to measure the differences between the actual rating and the predicted one.

In order to find out which classifier is the most appropriate one to use for this dataset and make a good prediction for the movie domain, the experiment was conducted on four different classifiers, detailed below:

4.6.1 The k-Nearest Neighbour (k-NN)

It is a classifier that find the *k* nearest neighbours. The given user is assigned to a similar user that shares the most common features of its *k* nearest neighbour users. Certain factors need to be considered, such as the similarity measurements, which calculate the distance between two vectors. With this classifier the Euclidean distance was utilised, as detailed in section 3.5.3.

4.6.2 Random Forest

It is an ensemble learning classifier that builds a set of decision trees. Each tree is developed from a bootstrap sample from training data. It is more robust with respect to noise (Breiman, 2001). This method has been successfully approved as an accurate machine learning classifier (Louppe, Wehenkel, Sutera & Geurts, 2013). The number of trees is utilised as 10, 20, 50 and 100, which are the most likely changes in this range (H. Zhang, Min & Wang, 2014).

4.6.3 Neural Network

This is a feed-forward neural network. This method has been successfully approved as an accurate machine learning classifier and for the experiments with 100 layers that have been used.

4.6.4 AdaBoost

This is another well-known classifier that can be used in machine learning applications. For the experiments, a learning rate of 1 has been used with 10 estimators.

4.7 Cross-validation

In order to test, validate and make a fair comparison between the proposed method and the baseline, cross-validation was used. This can help in assessing the quality of the results. The main goal of cross-validation in this thesis was to measure how accurately the predictive method that is proposed will perform in practice. A prediction problem was conducted for all experiments, which involves calculating the differences between the actual rating and the predicted rating. However, to make the proposed algorithms more robust, a different evaluation test was

conducted to reduce the bias and the variance (R Kohavi, 1995). This is detailed in the next section.

4.7.1 Leave-one-out

In this case, one element is removed from the training data, then the remaining data is constructed. After that, the removed element is tested. Shenghui Wang et al. (2014) utilise the leave-one-out cross validation to validate the influence of the recommendation accuracy.

Although leave-one-out gives an unbiased estimate of the expected generalisation error, and it is a good choice in the prediction domain, it is very costly to calculate the values since it takes a lot of time to run on the training algorithm (Chapelle, Vapnik, Bousquet & Mukherjee, 2002). Hence, with big datasets like those used in this thesis, such as MovieLens 1M, k-fold cross-validation is conducted instead, as explained below.

4.7.2 k-fold

The dataset is spilt into *k* folds that are trained and tested. Cross validation estimates the accuracy through calculating the correct classification divided by the number of instances in the dataset (R Kohavi, 1995).

4.8 Evaluation metrics

Recommender systems researchers have applied different measures to evaluate recommendation algorithms in terms of accuracy and quality. Since 1994 (Resnick et al., 1994), the accuracy of recommender systems has been evaluated in the literature in different ways. Most of the empirical studies examining recommender systems have focused on appraising the accuracy of these systems (Herlocker,

Konstan, Borchers & Riedl, 1999). This insight is useful for evaluating the quality of a system and its ability to forecast the rating for a particular item. There are two main types of accuracy measurement metrics for recommender systems:

1. Predictive accuracy metrics, which measure similarity between true user ratings and recommender systems' predicted ratings. This research applies accuracy metrics to measure the performance of the proposed methods. Both the mean absolute error (MAE) and the root mean squared error (RMSE) are used to evaluate the prediction accuracy of the different recommendation techniques. Prediction accuracy is enhanced when MAE and RMSE are lower.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - r_i| \tag{20}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - r_i)^2} \tag{21}$$

   In the above equations, p_i is the predicted rating, and r_i is the actual rating. The differences between them provides an assessment of how well the prediction algorithms match the actual rating (Konstan & Riedl, 2012). It should be considered that lower values indicate a better result in terms of accuracy.

2. Classification accuracy metrics. In this type of evaluation, precision, recall and f-measures are used to measure classification accuracy.

4.9 Summary

This chapter presented to the reader the developed algorithms that were used in each experiment. It introduced the settings for each experiment and provided details about the parameters used, such as the number of k, which represents the number of chosen neighbours, and the type of evaluation that was used in each experiment.

Moreover, details of the datasets that were used in this thaesis are provided. Finally, the prediction accuracy measures that were conducted to evaluate the experiments were introduced. In all the experiments the proposed algorithm showed an improvement, with an error rate lower than the baseline and the state-of-the-art recommendation methods.

5 Results and discussion

5.1 Introduction

To compare the proposed method with the baseline and state-of-the-art recommendation techniques, several algorithms were implemented. First of all, a novel switching method was tested that solves the long tail recommendation problem through relying on past users' experience to retrieve similar ratings for a new user. This is done by utilising both collaborative filtering and content-based filtering through a switching method to mitigate the increased error rate of recommender systems while dealing with low-ranked items in the tail. The low-ranked items are defined as unpopular items and the high-ranked items as popular. The proposed method utilises user history as a case base that helps in reducing the long tail recommendation problem through increasing the probability of identifying similar items from a user's past ratings of similar items. These cases were selected in the retrieval step using case-based reasoning to switch between CF and CBF as a hybrid recommender system method.

Secondly, the extended experiment was presented, a multi-level recommendation algorithm that applies a switching method. It improves prediction accuracy when recommending items in the long tail. This method was also examined through a comprehensive experiment on a real public dataset using two training/testing approaches to show the quality of the proposed method, conducting a comparison with the baseline methods and a state-of-the-art alternative.

Moreover, a new method was proposed that utilises a triangle similarity measure via a multi-level algorithm. In this method both the length and the angle of the rating vectors between users are applied, as well as the constraints that modify user

similarity, assigning these to different levels. An extensive experiment was conducted to show its effectiveness based on three real datasets through a comprehensive comparison with a baseline and a state-of-the-art alternative.

Finally, a hybrid feature combination method was tested using four different classifiers: k-Nearest neighbour (k-NN), Random Forest (RF), Neural Network (NN) and AdaBoost. The experiment includes the results that made a comparsion between the four classifier when demographical attributes is combined with the collaborative filtering approach.

The all experiments of this thesis was mainly focused on the predictive accuracy by calculating the error rates. In addition, various well known datasets were used in each experiment to prove the propsed algorithms are able to consider small and big datasets. beside that, to make a reliable results different neighorhood size was taking into account in each experiment. Moreover, since the propsed method relys on the featuers of items, movie genres were used for each dataset.

5.2 Switching CBR experiment results

This section presents and explains the results obtained in these experiments. The MovieLens dataset was used to evaluate the proposed method. The predictive accuracy rate was calculated in order to compare the improvement rate across all the algorithms. In order to evaluate the accuracy of each method, a leave-one-out method was applied. For each query, the difference between the predicted rating and the actual rating was calculated. This evaluation was executed with the following conditions:

5.2.1 Collaborative filtering

In this test, the accuracy rate was calculated using only a collaborative filtering method based on users.

5.2.2 Content-based

Next, the experiment was repeated with a content-based method based on users. This method uses the statistical average of ratings per genre defined in each user description.

5.2.3 Combined method

In this experiment, both recommender systems were combined with a similarity function. In this case, the similarity between two users is the average of the similarity calculated with the CF method and the CBF method.

5.2.4 CBR approach

Finally, the proposed system, which uses a switching method between CF and CBF, was tested, as explained in more detail in Chapter 3.

5.2.5 Experimental results

The results of the proposed method are examined with regard to predictive accuracy through a comparison with the baseline models. All tests were varied by systematically changing two elements: the similarity function used (less in the content-based model based on items) and the k in the k-NN algorithm. The similarity functions used are Euclidean distance similarity and Pearson similarity, as shown in equations (12) and (14). The values of k are 3, 5, 10, 15 and 25. Moreover, in order to find the optimal value of δ, a number of experiments were

performed by changing the value of δ from 5 to 20 with a difference of 5. In each set of results the MAE was observed, and it was found that the optimal number is 10 as shown in Figure 12. Hence, all experiments were conducted using 10 as an optimal switching criteria.

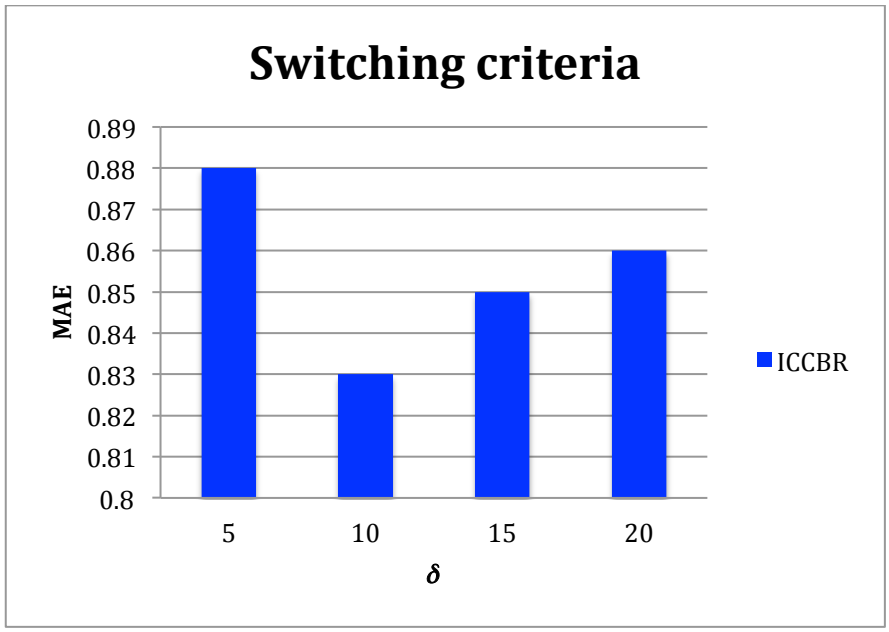

Figure 12 MAE results with different value of δ

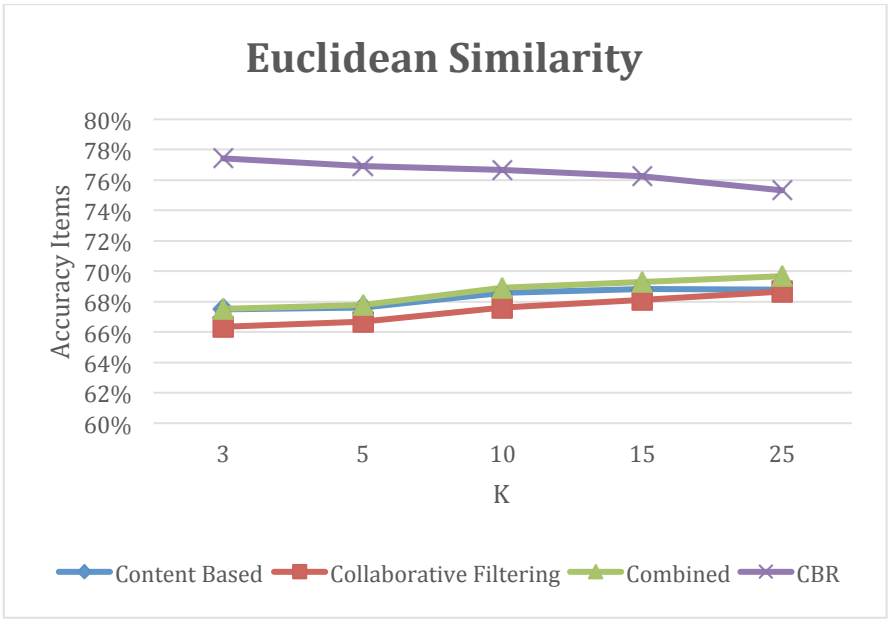

Figure 13: Accuracy rate based on Euclidean distance similarity

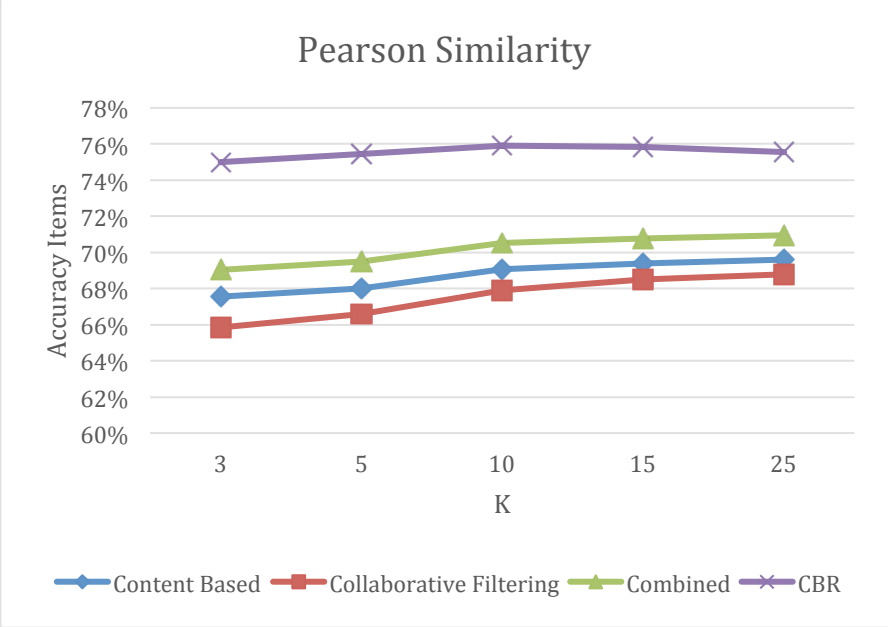

Figure 14: Accuracy rate based on Pearson similarity

Figure 13 and Figure 14 above show the accuracy rate of each method using different similarity functions, and different values of *k* in the k-NN algorithm. These results show that the CBR method presented in this experiment improves the general accuracy of prediction for both functions. It can be seen that the accuracy of the other methods is less than 70% for both similarity functions. The proposed method improves accuracy by between 6 to 10% depending on the similarity function and the *k* selected in the k-NN algorithm.

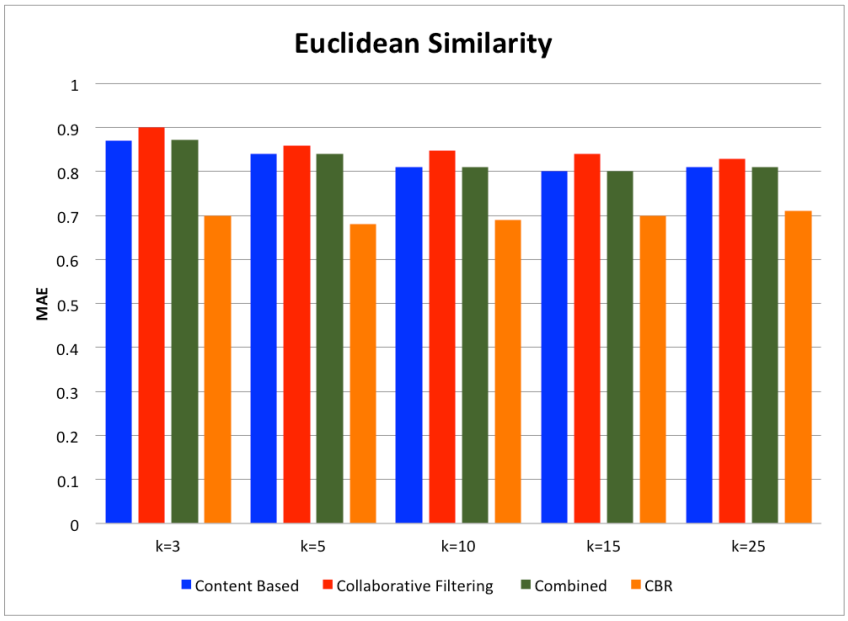

Figure 15: MAE base on Euclidean distance similarity

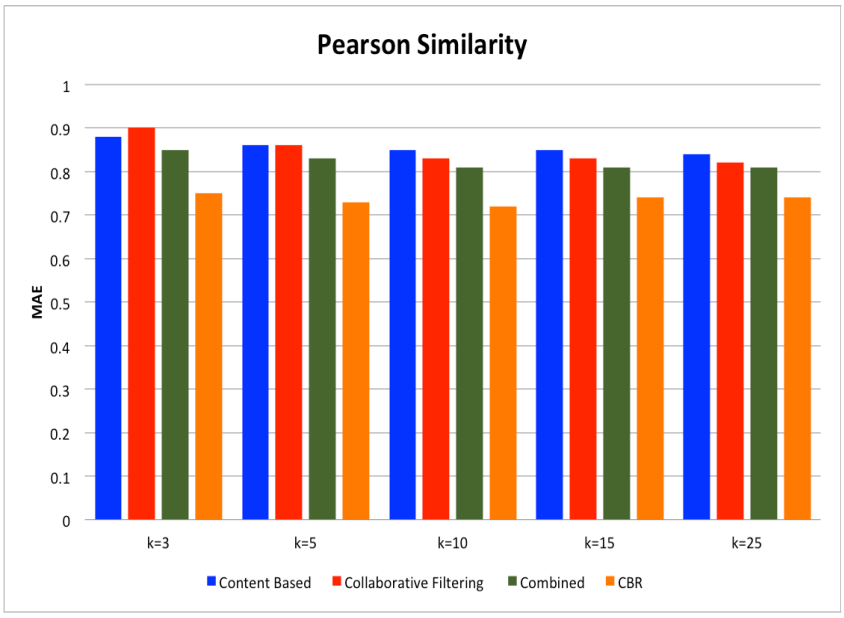

Figure 16: MAE based on Pearson similarity

Figure 15 and Figure 16 show the performance of the method. They also show the mean absolute error (MAE) of each method. They demonstrate that the error rate of the proposed CBR system is lower than that of other approaches. For example, the CBR system is better able to predict unknown item ratings in the MovieLens dataset.

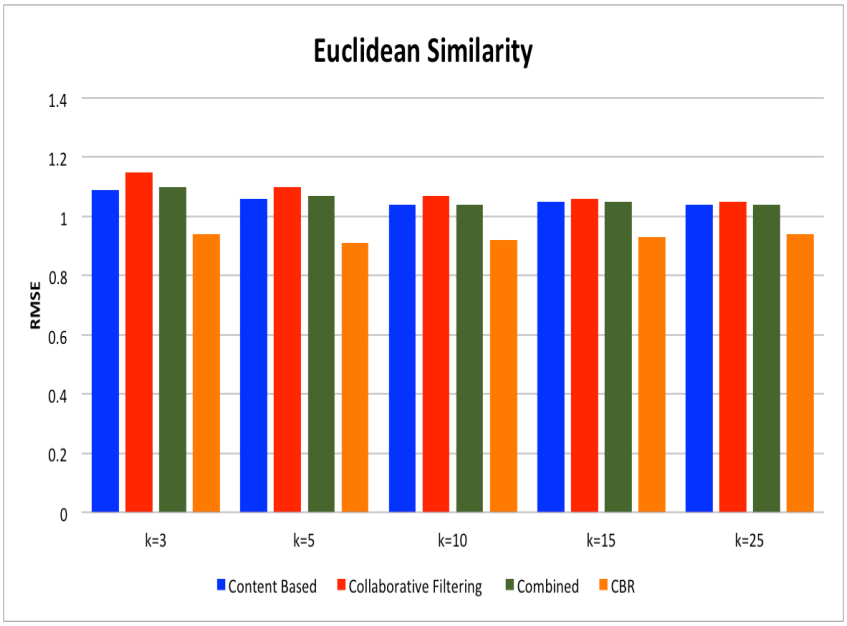

Figure 17: RMSE based on Euclidean distance similarity

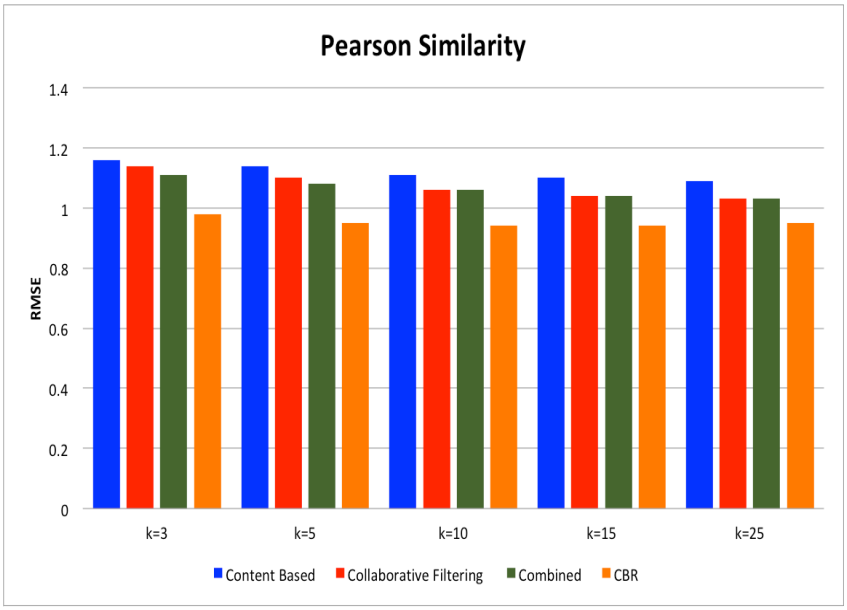

Figure 18: RMSE based on Pearson similarity

As can be seen in Figure 17 and Figure 18, the root mean square error (RMSE) demonstrates that the performance of the system is superior to that of the baseline methods.

To calculate the improvement rate of the new system, the improvement function proposed by Park and Tuzhilin (2008) was used. This calculates the improvement rate based on a comparison between the new system and the baseline.

$$Improvement\ rate = \frac{RMSE_{base} - RMSE_{CBR}}{RMSE_{base}} \quad (22)$$

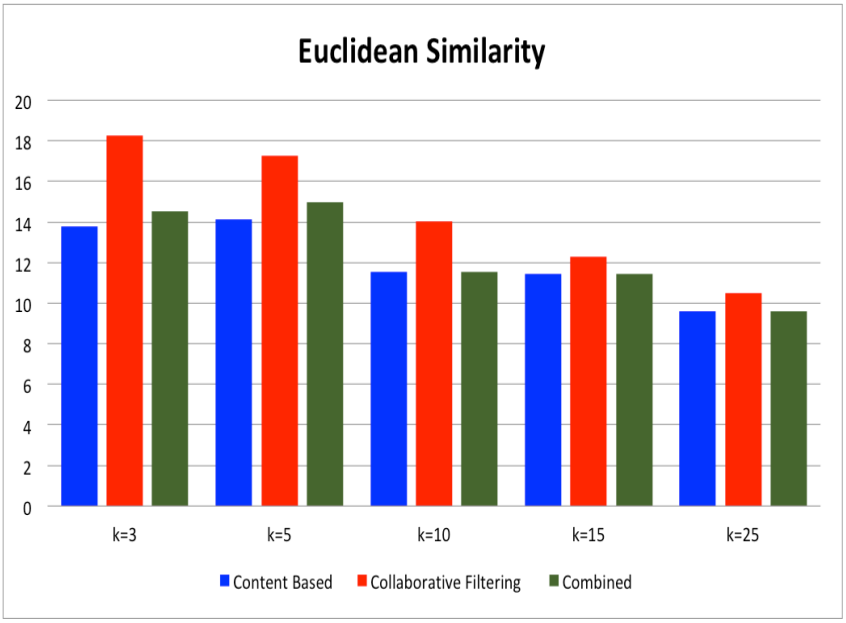

Figure 19: Improvement rate for the CBR system based on Euclidean distance similarity

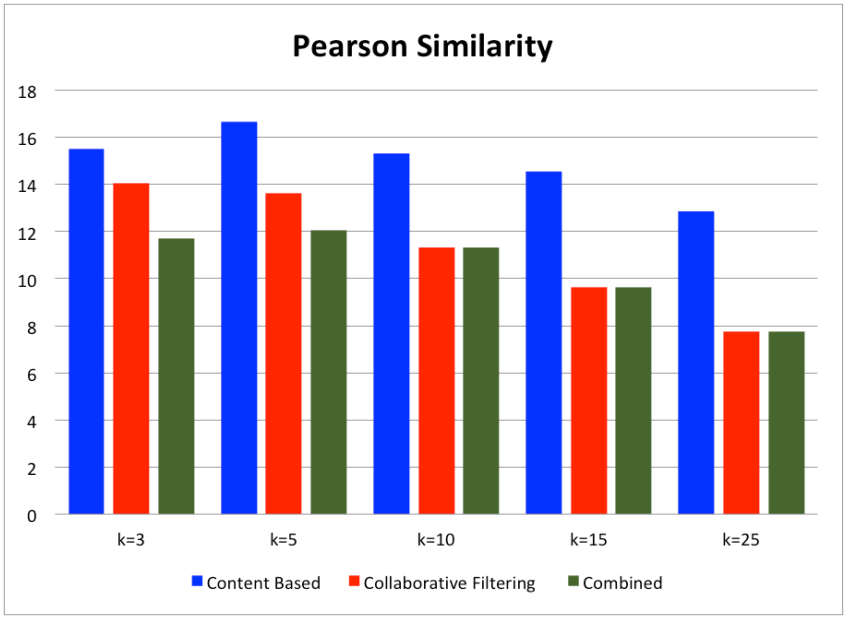

Figure 20: Improvement rate for the CBR system based on Pearson similarity

Figure 19 and Figure 20 compare the improvement rate of the CBR system with the baseline. Those figures show that the proposed method improves prediction by between 8% and 18% using the Euclidean distance similarity function. In addition,

using the PCC similarity function, the CBR system improves the rating by between 8% and 16%.

A novel method that solves the long tail recommendation problem was examined using a case-based reasoning system. When the system is asked for a new item (a movie), it will obtain the number of ratings received by this movie. If the movie does not have a sufficient number of ratings, then this movie belongs to the long tail problem. Hence, it will use the switching CBR approach to calculate the rating. In this case, the CBR system retrieves the cases based on the user's history of movies they have rated in the past.

5.3 Switching multi-level experiment results

This experiment is based on a switching method that integrate the multi-level algorithm with a number of constraints. The constraints provide a higher similarity value between the users that have more common items in which the similarity value is above a certain threshold, which is something that is not available in other methods. However, it should be noted that the similarity value between the users is set to be $Sim_{i_j i_q}^{PCC}$, when the result does not meet one of the defined levels. This is different from the equation proposed by Polatidis and Georgiadis (2016), where it is set to be 0. The changes of the equation is made after observing the results and it was found that in many cases the similarity was assigned to 0 as it is shown in figure 21 and 22 .

```
user1,user2,similarity
1.0,1.0,1.25
1.0,2.0,0
1.0,3.0,0
1.0,4.0,0
1.0,5.0,0
1.0,6.0,0
1.0,7.0,0
1.0,8.0,0
1.0,9.0,0
```

Figure 21 the observed similarity results using Polatidis equation

```
user1,user2,similarity
1.0,1.0,1.25
1.0,2.0,-0.0028281478205093 76
1.0,3.0,-0.0023166591017492 63
1.0,4.0,0.042864412532782 2
1.0,5.0,-0.0034258978882044226
1.0,6.0,-0.0021138865598289 18
1.0,7.0,0.063292891356298 44
1.0,8.0,-0.0038515939427503 45
```

Figure 22 the observed similarity results using the propsed equation

This experiment was conducted to compare the proposed method's results with the baseline CF, CBF algorithm and the switching method with the tradition similarity that is examined in section 5.2. Hence, CBF means the content-based filtering that relies on average of ratings per genre defined in each user description. CF_Euclidean is the collaborative filering method that calculate the similarity between users using Euclidean distance. CF_Pearson apply the same collaborative filtering method but with Pearson correlation measuere. Finally, ICCBR is the switching method that is propsed in section 5.2.

This experiment is based on the latest publicly available MovieLens dataset and the 1 milion movieLens dataset. It is evaluated using most popular accuracy measures

for predictive accuracy in recommender systems, mean absolute error (MAE) and root mean squared error (RMSE). Given below are the results for the real datasets using two different training and testing percentages to evaluate the results and compare it with all methods. In this experiment, k represents the number of neighbours, specified to be equal to 3, 5, 10, 20 and 30.

5.3.1 Experimental results

Figure 23 and Figure 24 show the MAE and the RMSE rates respectively across the MovieLens 100k dataset utilising 70% for training and 30% for testing. It is shown that the proposed method outperforms all the other methods. It can be clearly seen that when the number of neighbours is smaller, for example, when $k = 3, 5$ and 10, the improvement is very significant. On the other hand, when k is higher the improvement in the proposed method is less effective.

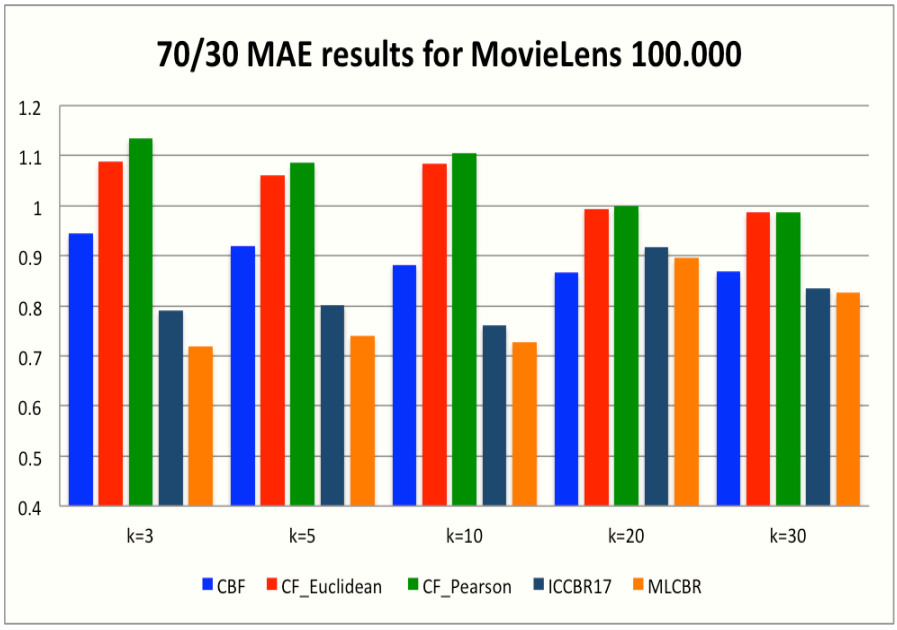

Figure 23: MAE results for MovieLens dataset using 70/30 test

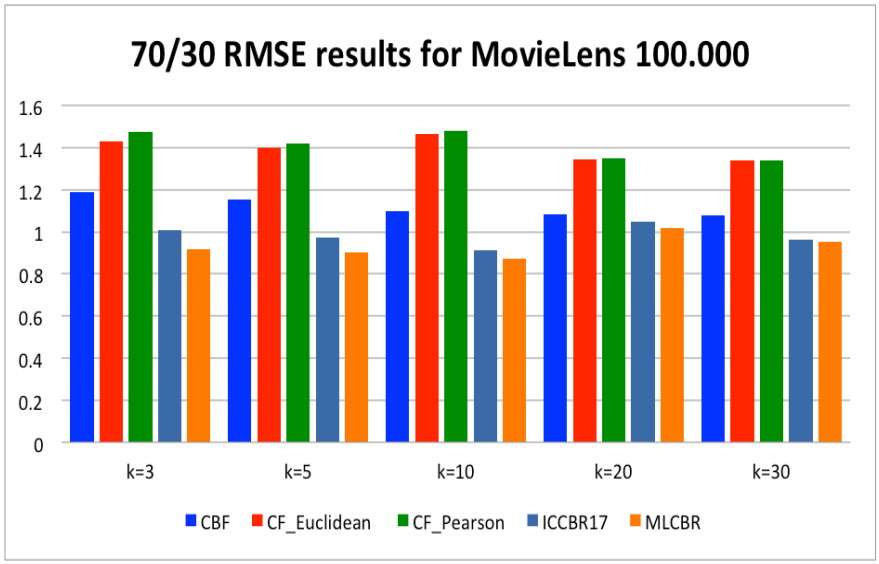

Figure 24: RMSE results for MovieLens dataset using 70/30 test

Figure 25 and Figure 26 also show the MAE and the RMSE rates over all methods using 60% of the dataset for training and 40% for testing. It can be seen that MLCBR achived the best MAE results with $k = 3, 5, 10, 20$ and 30. More precisely, it is noticeable that when k is small, the improvement is higher as also shown in the experiment with 70% training. On the other hand, the user based collaborative filtering shows the worst results in both similarity functions: Euclidean and Pearson.

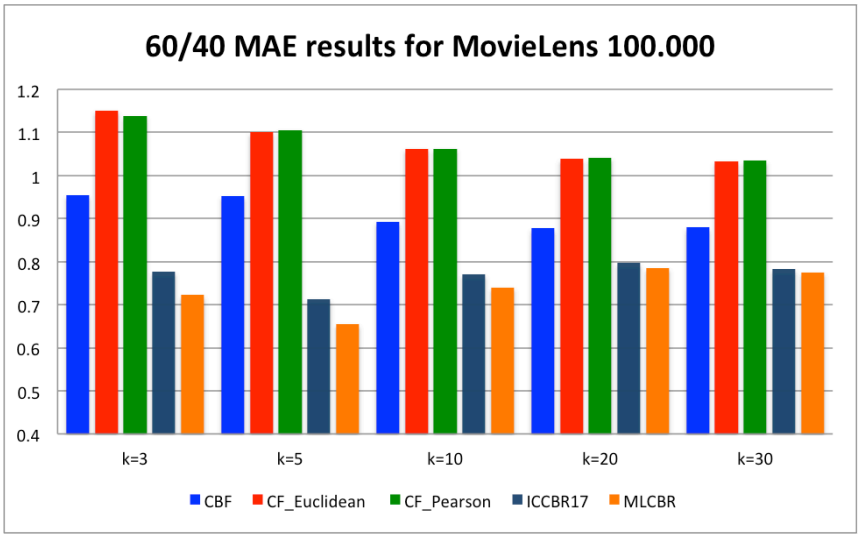

Figure 25: MAE results for MovieLens dataset using 60/40 test

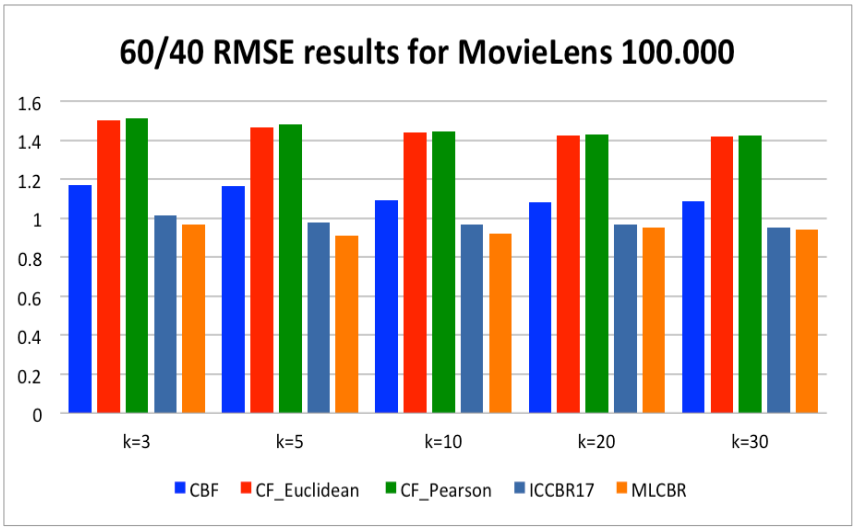

Figure 26: RMSE results for MovieLens dataset using 60/40 test

Figure 27 shows the comparison of the improvement rate of the proposed method compared with the baseline in the results obtained. This figure shows that the prediction accuracy is improved by between 36% to 40% using the collaborative filtering method with Euclidean and Pearson similarity measures. In addition, there is an improvement compared with content-based filtering between 21% to 25%. Finally, the switching method using traditional similarity measures is improved by 1% to 5%.

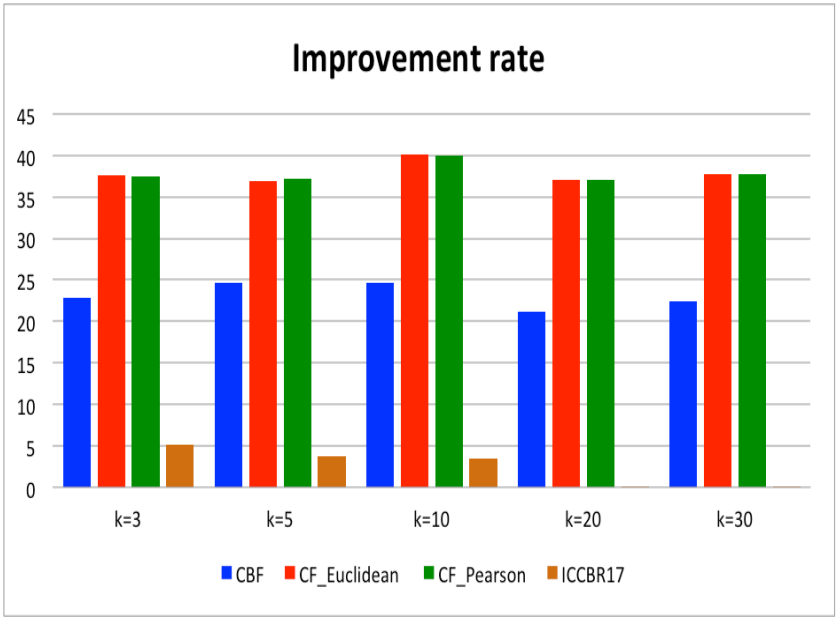

Figure 27: The improvement rate over all methods

Figure 28 and Figure 29 show the MAE and RMSE rates respectively across MovieLens 1M dataset using the aforementioned predictive methods utilising 70% for training and 30% testing. In addition, Figure 30 and Figure 31 use 60% for training and 40% for testing. It is shown that the proposed method outperforms all the other compared recommendation methods. It can be seen clearly that when the number of neighbours is smaller, for example, when $k = 3, 5$ and 10, the improvement is very clear. On the other hand, when k is getting higher the proposed method still improves but it is less effective.

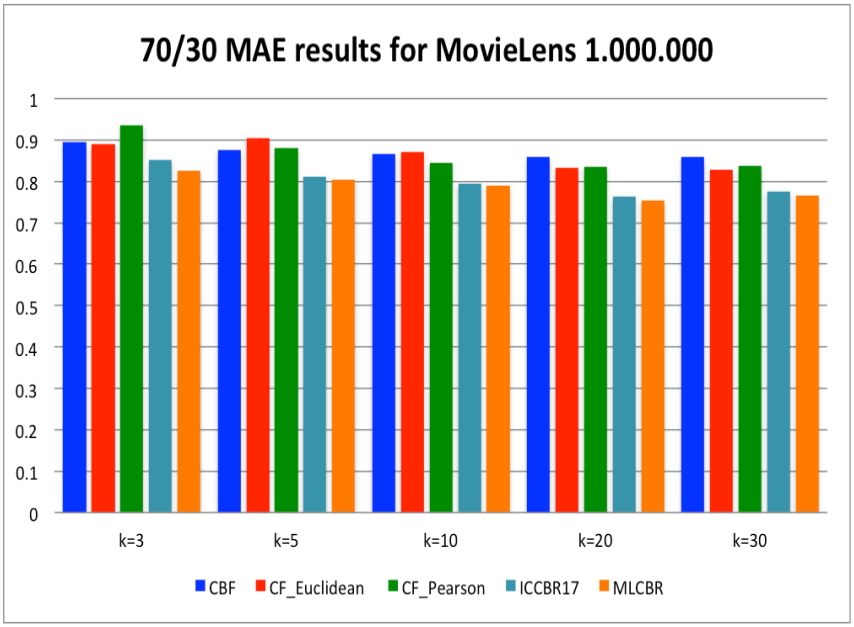

Figure 28 MAE results for the MovieLens 1.000.000 dataset

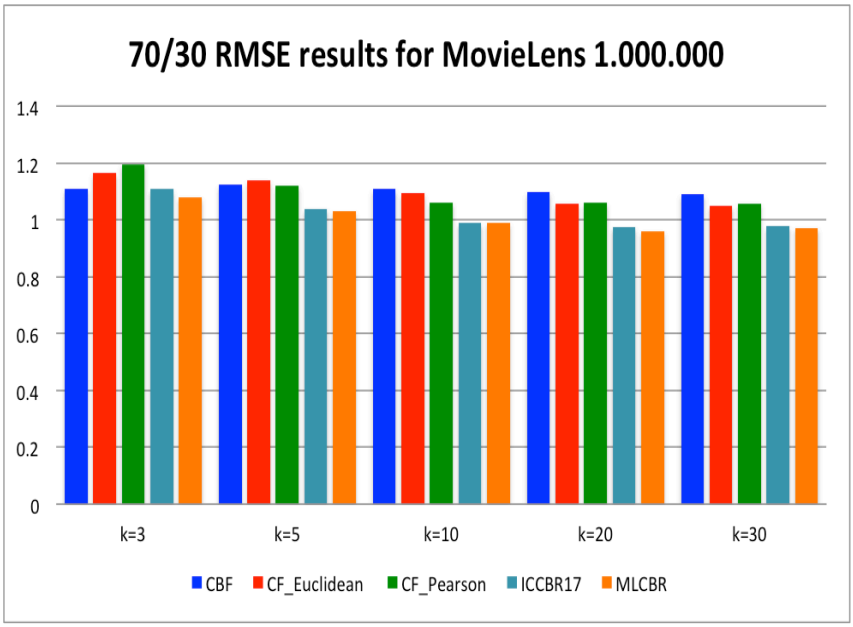

Figure 29 RMSE results for the MovieLens 1.000.000 dataset

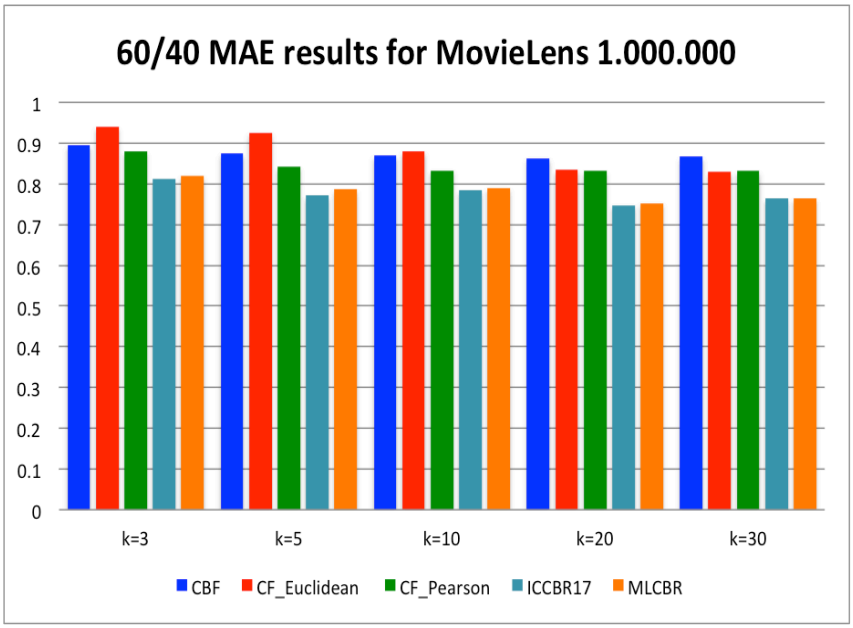

Figure 30 MAE results for the MovieLens 1.000.000K dataset

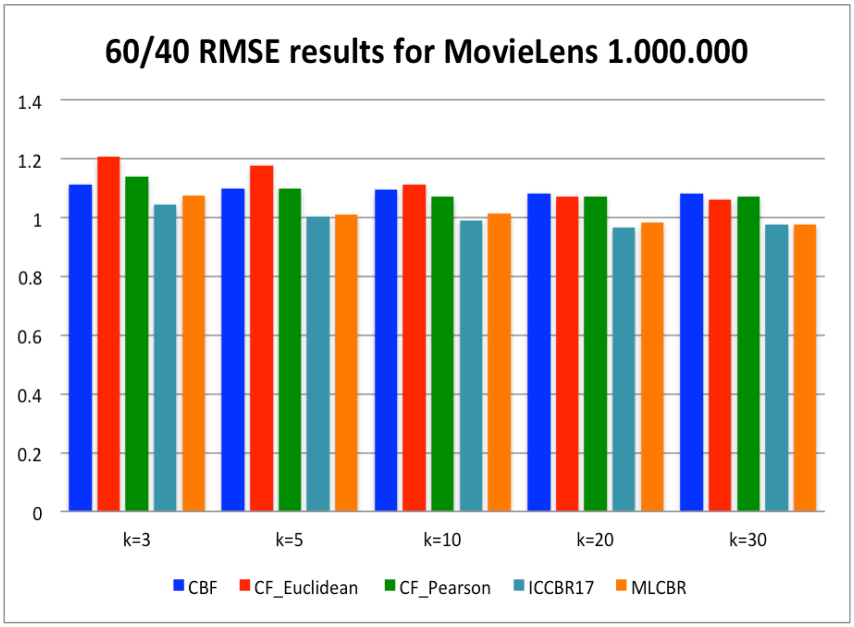

Figure 31 RMSE results for the MovieLens 1.000.000 dataset

The comparison of the improvement rate for MAE of the proposed method compared with the baseline in the results obtained using MovieLens 1.000.000 dataset is shown in Figure 32. This figure shows that the prediction accuracy is improved by between 7% to 12% using the collaborative filtering method with Pearson similarity measures. In addition, there is a significant improvement compared with collaborative filtering using Euclidean distance similarity measueres between 7% to 11%. Moreover, the content-based filtering is improved by between 21% to 25%. Finally, the switching method using traditional similarity measures is improved by 1% to 3%.

Additionally to that, Figure 33 presents the improvement rate for the MovieLens 1.000.000 dataset. It is shown in comparison that the prediction accuracy is improved when the proposed method is being used both against the baselines and the ICCBR17.

(23)

$$Improvement\ rate = \frac{MAE_{base} - MAE_{Switch}}{MAE_{base}}$$

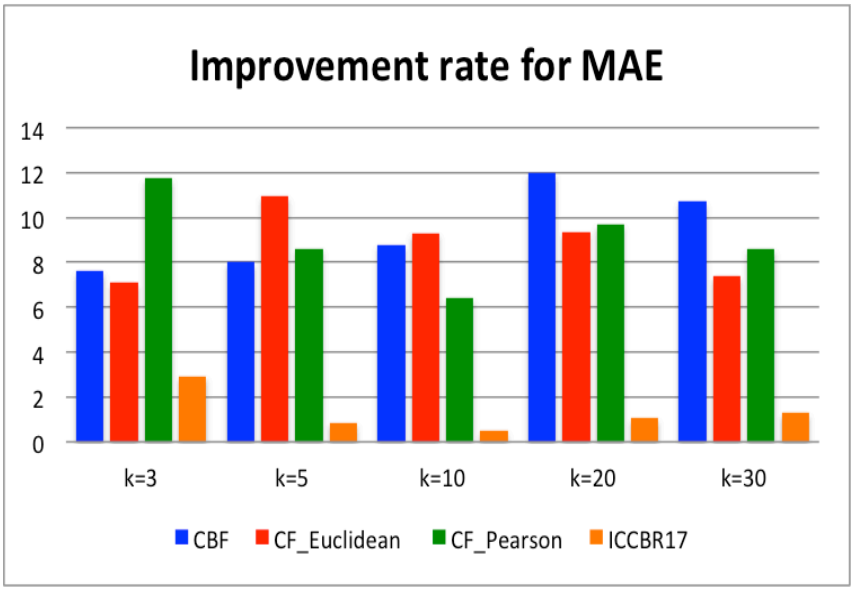

Figure 32 The MAE improvement rate for movieLens 1.000.000 dataset over all methods

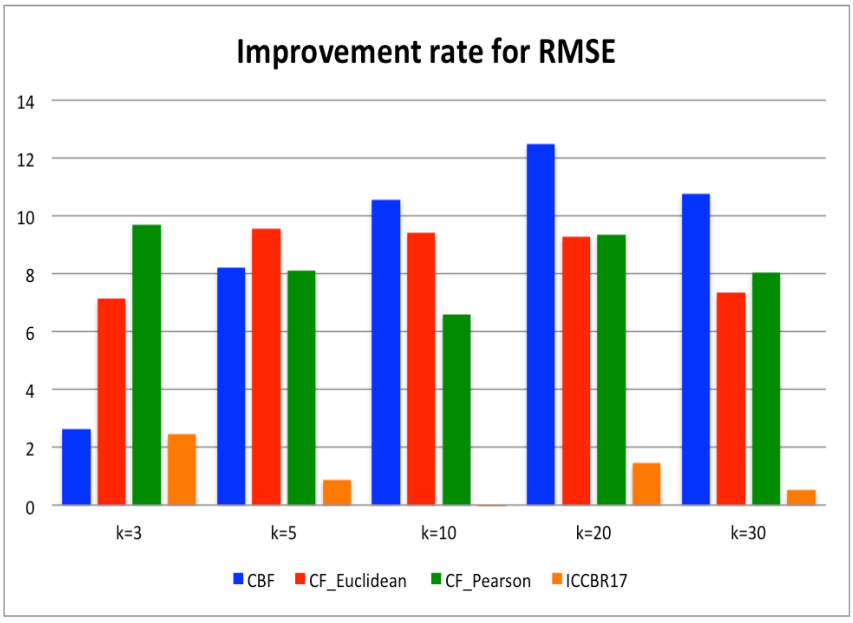

Figure 33 The RMSE improvement rate for movieLens 1.000.000 dataset over all methods

5.3.2 Discussion

The long tail recommendation problem becomes a problem and opportunity at the same time when a point has been reached within a business where offering the same popular products or services to users is not a viable business strategy anymore. This might be the case if the users might not want to select popular products all the time but want to broad their taste. By offering a recommendation method that provides more accurate recommendations of items found in the long tail a solution can be provided to users that we want to be loyal to a business by providing an effective recommender system that includes items from the long tail as well or in a total different scenario this could be recommending news content to users that it's difficult for them to find elsewhere.

A typical business model might usually include different types of recommendations such as (a) popular items (b) what users with similar purchase history like (c) content-based (d) long tail and many other possible combinations as well. In this experiment, the development of a recommendation method is concentrated that improves the accuracy of the recommendations in the long tail, which is considered to be an important business model to the recommendation process. The proposed recommendation method fits this business model as it provides more accurate recommendation to users or to explain this more accurately possible customers of e-commerce or other relevant websites such as social networks. The proposed method has been evaluated using two real datasets with the results validating it in most scenarios and it has been compared against a number of alternative methods used as baselines. Moreover, the well known error rating prediction metrics MAE and

RMSE have been used along with an overall improvement rate for each of the metrics and datasets.

The proposed recommendation method can assist further the business model of websites by increasing the user experience. This is important since for increasing sales that meet users need, then they will like a service and will come back to use it again. Furthermore, the search burden of users will be low since more relevant products or services will be available without searching for them and the processing power required by a vendor will be reduced, resulting in a win-win situation for both the customers and the vendors. Additionally to the aforementioned, a good quality recommendation method for the long tail can become a good social opportunity as well where a diversity of products, manufacturers and contents is good for developing a marketplace where is possible for a diverse set of products to find potential customers. Recommending in the long tail is a way to help avoiding recommendations from a well known list of products only.

5.4 Triangle similarity with multi-level algorithm results.

Many collaborative filtering techniques have been proposed in different domains, such as e-commerce. Typically, elaborate approaches outperform the commonly used k-NN baseline method in terms of accuracy, particularly for sparse datasets or in terms of scalability as they rely on pre-processing or model-building phases (Gedikli & Jannach, 2010).

Most CF approaches analyse user ratings to determine the similarity between users and items. The similarity measure is important for finding accurate results in recommender systems. However, it is challenging to determine distance measures

in these systems in order to find similarities between users. Collaborative filtering is the most common applied algorithm through the k-NN approach (Herlocker et al., 2004b). The key issue in this technique is how to calculate the similarity between users or items by finding similar shared interests. It is relies on the rating aspect, which allows users to assign a high or low rating to a certain item based on their preference or dislike for it (Konstan & Riedl, 2012). Many similarity measures have been adopted in recommender systems, such as Pearson's correlation coefficient (PCC) (Resnick et al., 1994) and cosine (Shi et al., 2014) to provide recommendations based on absolute ratings between users.

Given below are the results of three datasets with different parameters, explained in detail in section 4.4. All algorithms were implemented in Java. In this experiment, *k* represents the number of nearest neighbours.

In this experiment the following item-based collaborative filtering IBCF algorithms was considered and compared with the following methods to evaluate the effectiveness of the triangle multi-level method and understand how this method can improve the predictive accuracy. All the methods utilised are detailed in the following sections:

5.4.1 PCC

In this method, the statistical correlation between the similar ratings of two users is calculated to find the users that are closest to a target user. The output will be a value between -1 and 1: 1 is a totally positive correlation, 0 indicates that there is no correlation and -1 is a totally negative correlation.

5.4.2 Multi-level CF

Multi-level CF is based on multiple levels, from top to bottom, with each of these levels having a number of constraints that is defined in Equation (1).

The two aforementioned methods were compared with the proposed method, which considers both the length and the angle of the rating vectors between users using triangle similarity. The multi-level algorithm also considers the right level for each user after calculating the similarity and compares it with the threshold. In addition, at each level, a certain constraint was conducted to modify the similarity between certain users who share similar items through co-rated items.

5.4.3 Experimental results

This section presnts the predictive accuracy results that have been achived by taking into consideration three different datasets. All figures show the error rates using the item-based collaborative filtering method. Furthermore, all experimnts were utilised by considering the k-Nearset Neighbour algorithm with different k such as k = 3, 10, 30, 50 and 100.

The MAE results in Figure 34 show that when the number of neighbours is small, prediction is significantly improved. For example, when $k = 3$ the $pcc = 1.002$; in the multi-level CF it is 0.83 whereas in the proposed method it is = 0.82. In addition, the proposed method outperforms the other methods in all cases, even when the number of neighbourhoods is high. In addition, It can be seen in Figure 35 that the results using RMSE are significant in the proposed method at $k = 30$ and $k = 50$. In contrast, the item-based collaborative filtering using Pearson similarity has the worst MAE results in all cases. Moreover, it is noticeable that when

performing the Pearson similarity measure the error rates are significantly higher compared to all methods.

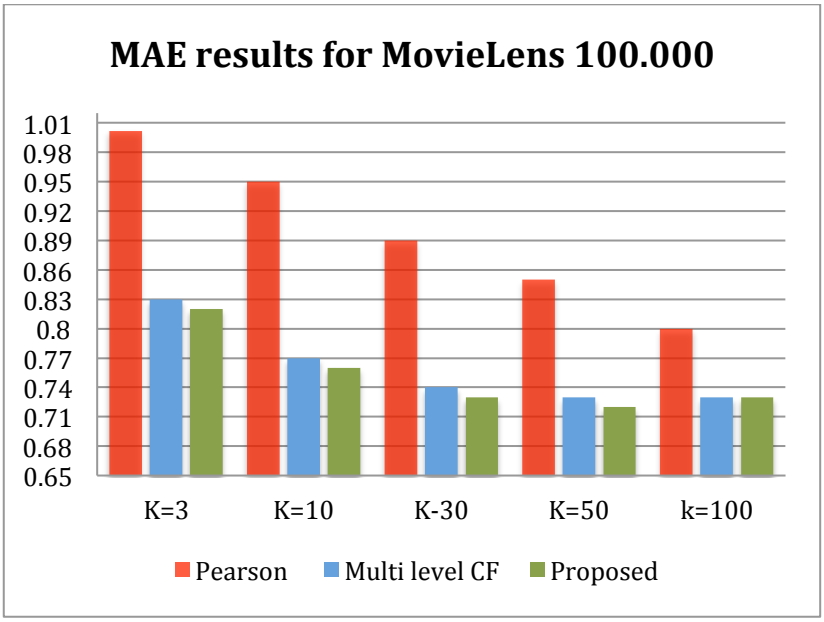

Figure 34: MAE results for the MovieLens 100K dataset

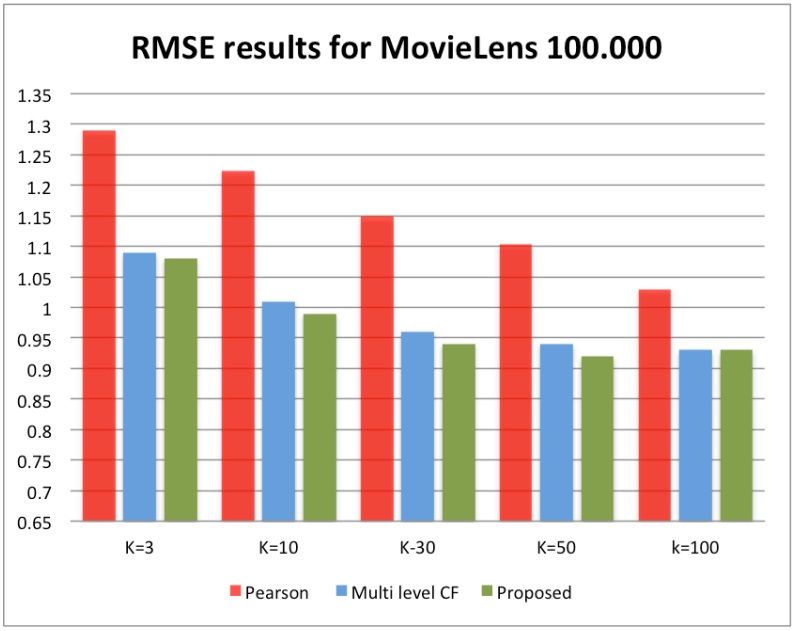

Figure 35: RMSE results for the MovieLens 100k dataset

Figure 36 and Figure 37 show the MAE and RMSE results using MovieLens 1M ratings. In these figures, the improvement of the proposed method compared with the others is clear. However, when the number of neighbourhoods is greater than

50, the multi-level CF results become very close or similar to the results from the proposed method, for example in $k = 100$. Overall, the proposed method achieves the minimal MAE in all cases.

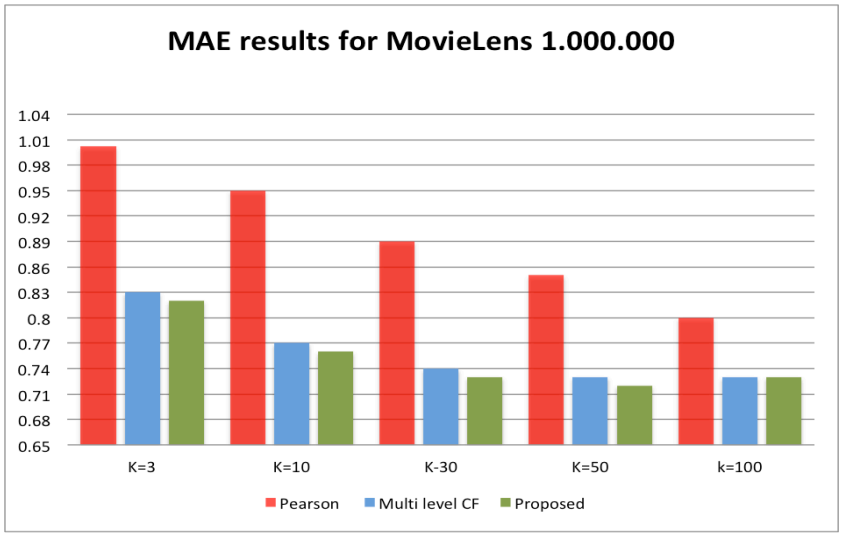

Figure 36: MAE results for the MovieLens 1M dataset

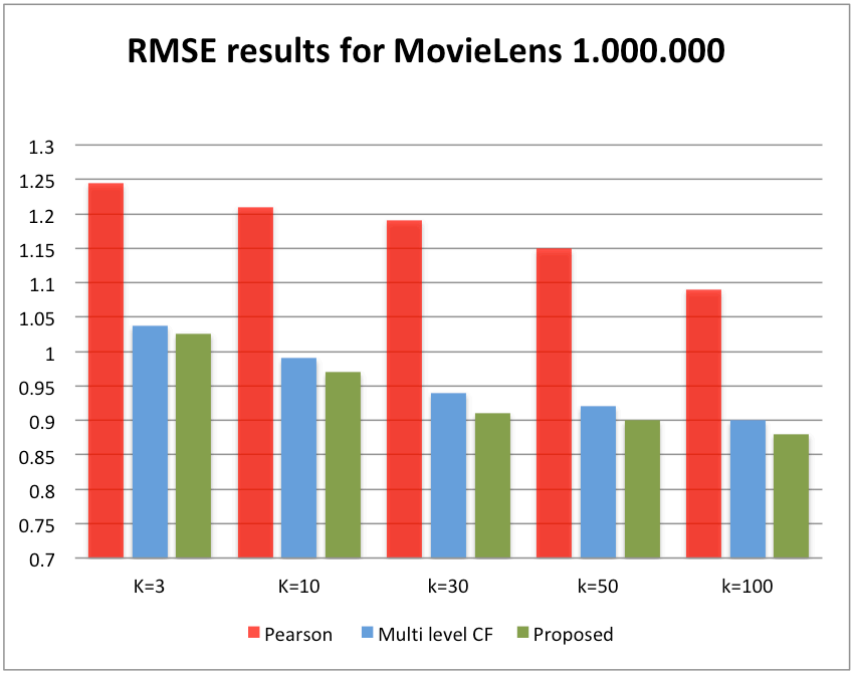

Figure 37: RMSE results for the MovieLens 1M dataset

For the Yahoo! Movies dataset, shown in Figure 38 and Figure 39, it can be seen that there is a significant change in all examined k values. For example, when $k = 3$, the baseline value is 0.85, the multi-level CF is 0.79 and the value of the proposed method is 0.76. However, when $k = 100$ it can be seen that the difference between the three methods is slightly smaller, but the proposed method still has the best results.

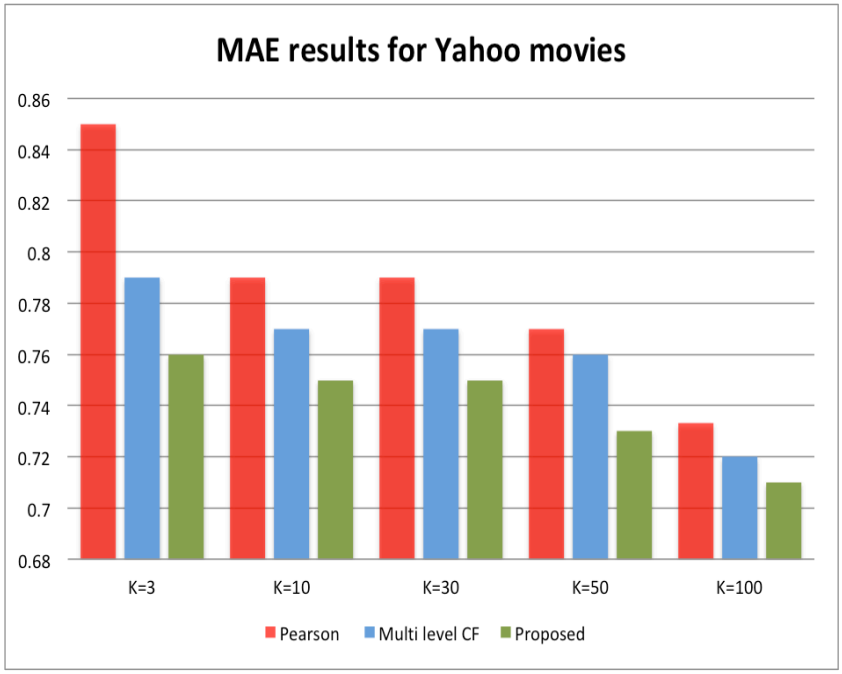

Figure 38: MAE results for Yahoo! Movies dataset

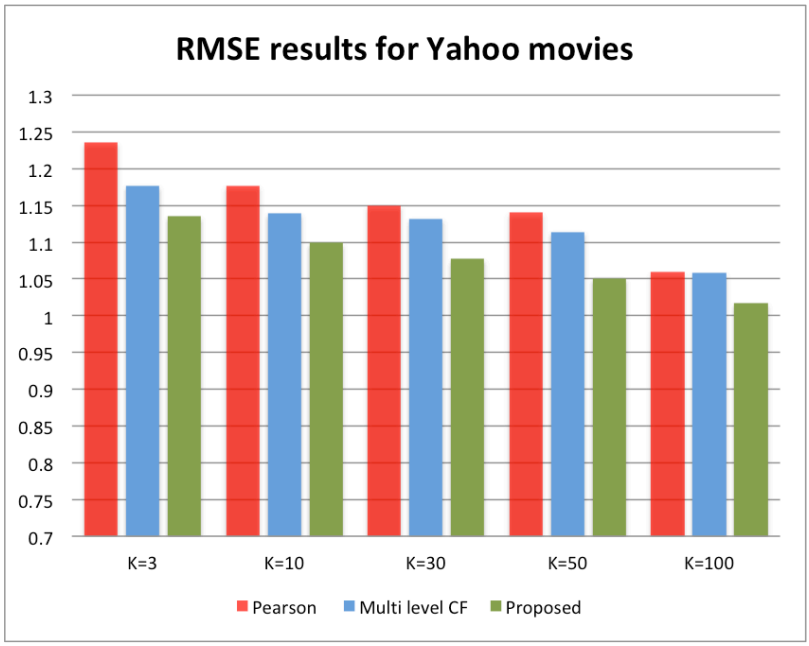

Figure 39: RMSE results for Yahoo! Movies dataset

Figure 40 shows the comparison of the improvement rate of the proposed method compared with the baseline in the results obtained using MovieLens 100.000 dataset. It can be seen clearly that the prediction accuracy is significantly improved by between 9% to 19% using the collaborative filtering method with Pearson similarity measures. In addition, there is a less effective improvement compared with multi-level CF between 1% to 3%.

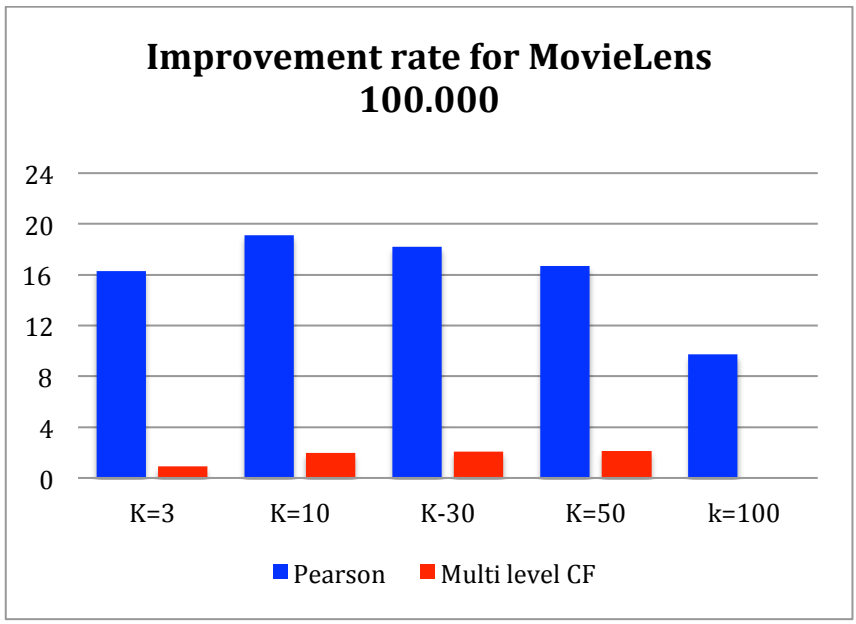

Figure 40 The improvement rate for movieLens 100.000 dataset over all methods

Figure 41 shows the comparison of the improvement rate of the proposed method compared with the baseline using MovieLens 1.000.000 dataset in the results obtained. This figure shows that there is a significant improvement compared with the collaborative filtering method with Pearson similarity measures by between 17% to 23%. In addition, there is a improvement compared with multi-level CF between 1% to 4%.

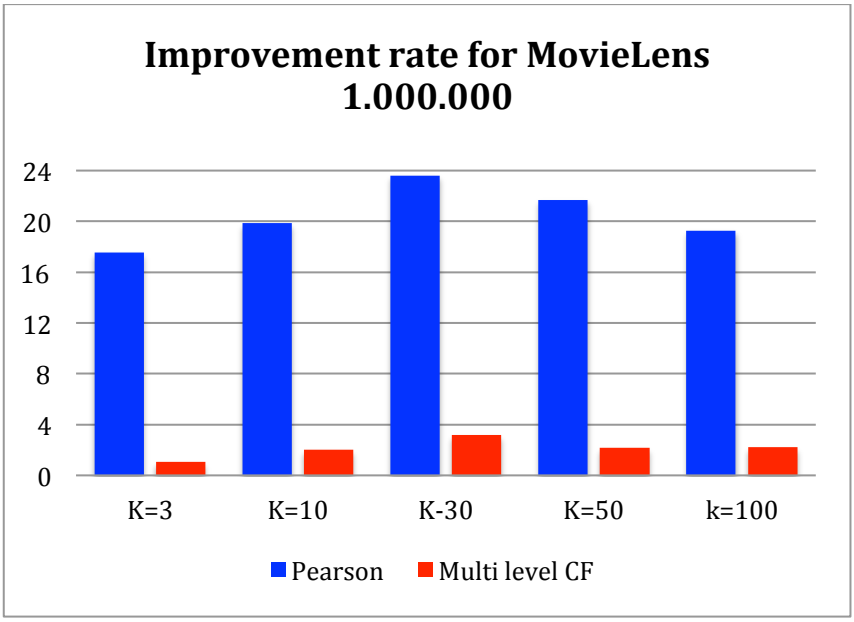

Figure 41 The improvement rate for movieLens 1.000.000 dataset over all methods

Figure 42 presents the comparison of the improvement rate for RMSE across Yahoo Movie! dataset . it is shown that the proposed method is improved compared with the baseline and against multi-level CF. This figure shows that the prediction accuracy is improved by between 4% to 8% using the collaborative filtering method with Pearson similarity measures. In addition, there is an improvement compared with multi-level CF between 3% to 6%.

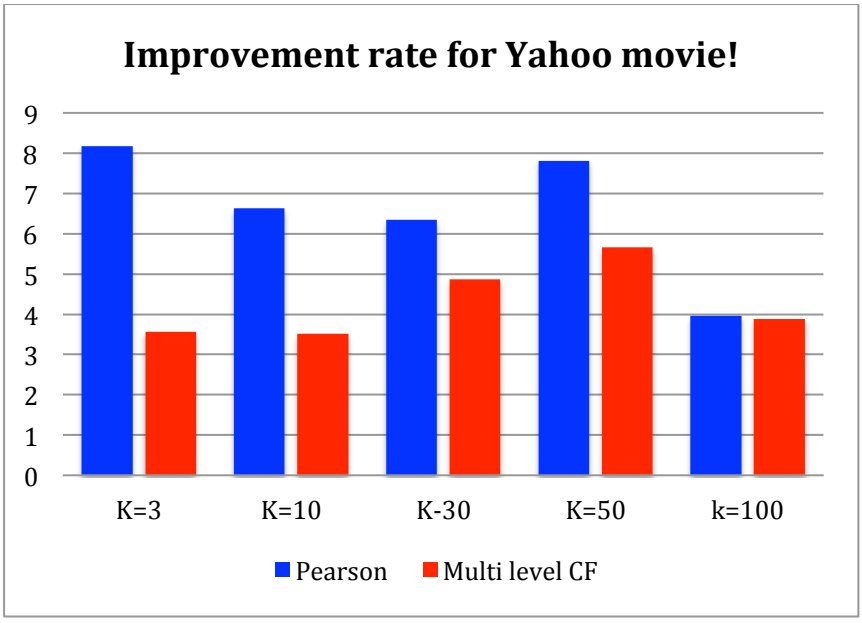

Figure 42 The improvement rate for Yahoo movie! dataset over all methods

5.4.4 Discussion

In this section, a novel item-based collaborative filtering method is proposed, based on triangle similarity and a multi-level similarity algorithm. The proposed method has been experimentally evaluated using three real datasets and well-known prediction accuracy metrics, with the results significantly outperformed the baseline methods. In the MovieLens datasets it is shown that the proposed method marginally outperforms the alternative, while in the Yahoo! Movies dataset the difference is higher between the proposed method and the alternatives. Furthermore, it is noticeable that the prediction error becomes smaller for each of the methods and in every dataset as the neighbourhood grows, with the difference between the proposed method and the alternative being similar in all cases. The proposed method outperforms the Pearson baseline and the multi-level CF state-of-the-art approach in all three datasets using MAE and RMSE.

In conclusion, this thesis has proposed a novel item-based collaborative filtering method that improves accuracy through solving the problem of sparsity of item distribution and the long tail problem. Compared to the benchmark recommender system methods, the proposed method obtains better results in terms of reducing the error rate of the prediction. Moreover, the results prove the effectiveness of the propsed method with the lowest error rate in all cases with different k and sevral datasets.

5.5 A demographic feature combination using k-NN and Random Forest

In this experiment, a novel hybrid feature combination method is proposed that combine user-based CF with the attributes of DF to indicate the nearest users, and compare four classifiers against each-other. This method has been developed through an investigation of ways to reduce the errors in rating predictions based on users past interactions, which leads to improved prediction accuracy in all four classification algorithms. An offline evaluation is used to test the approach and compare the results.

5.5.1 Experimental results

The results show the accuracy of the predictive method. As can be seen in Figure 43, the MAE accuracy metrics were made through applying different *k*, with $k = 3, 10, 30, 50$ and 100. Then the experiment was performed with the Random Forest classifier using different set of trees, as shown in Figure 44. Moreover, Figure 45 shows the results for the neural network and AdaBoostclassifiers.

It is clear that the performance is improved when the demographic features are combined at (DF+CF) in both classifiers. However, it is noticeable that the improvement in the AdaBoost classifier is significantly higher than all other classifiers. For example, in k-NN when $k = 3$ the improvement rate is only 1%, and in Random Forest, when $T = 10$ the improvement is 5% whereas in AdaBoost the improvement is 10%.

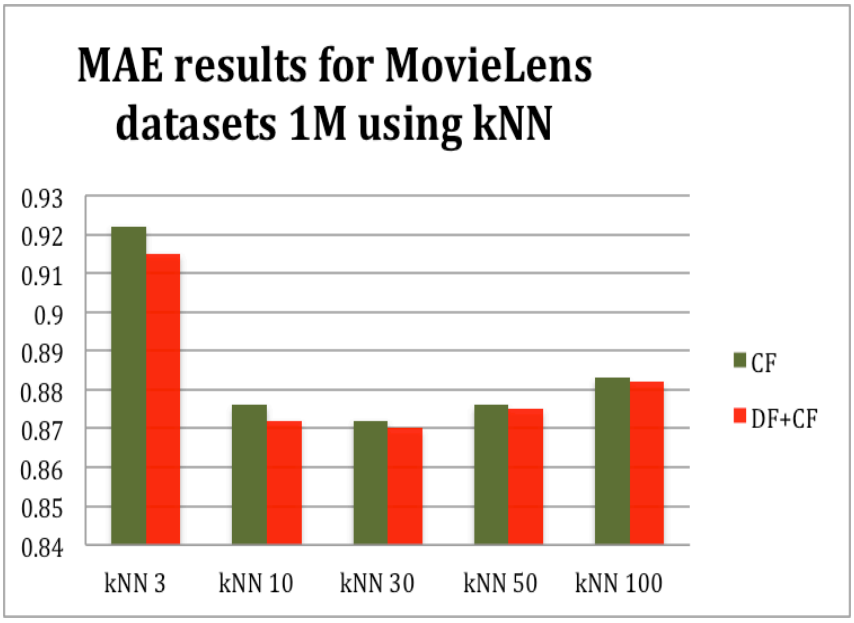

Figure 43: MAE results for the MovieLens dataset using k-NN

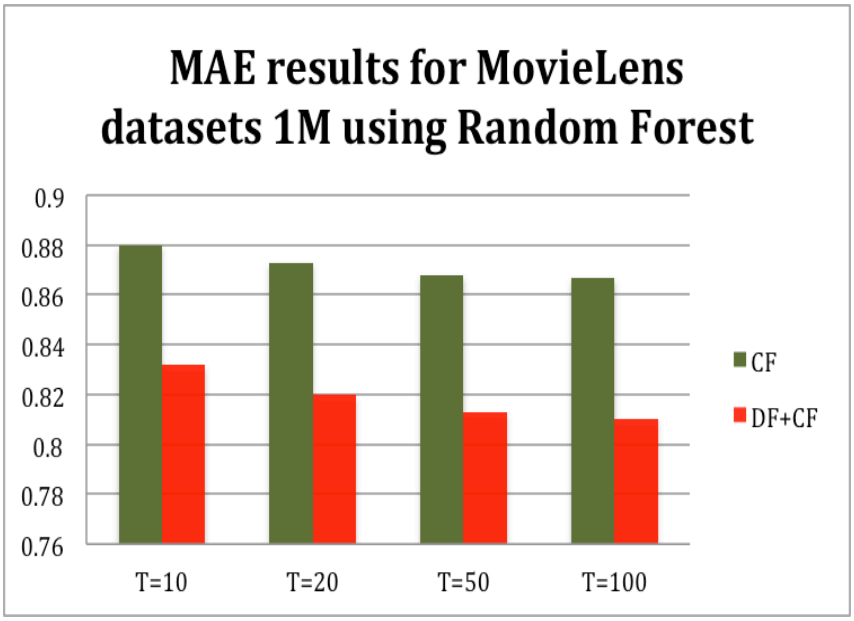

Figure 44: MAE results for the MovieLens dataset using Random Forest

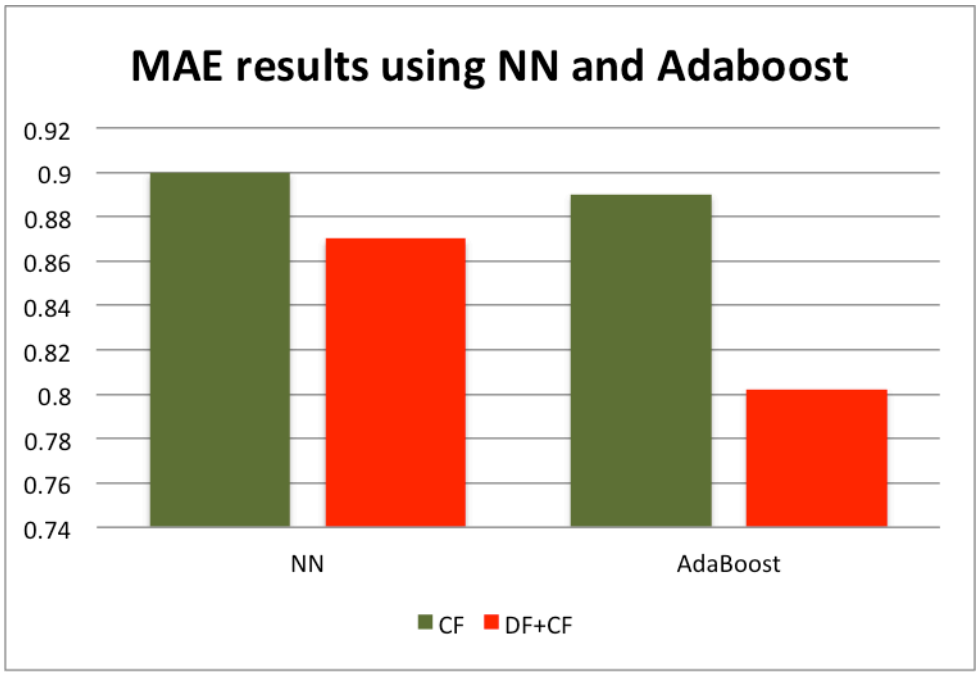

Figure 45: MAE results for the MovieLens dataset using NN and AdaBoost

Figure 46, Figure 47 and Figure 48 show the results using RMSE. There is a significant improvement in the AdaBoost compared to k-NN, Random Forest and NN in all sets. Foe example, in AdaBoost classifier the result is 1.3 when collaborative filtering is used. In contrast, the result of the combined method is 1.1.

Additionally, in Random Forest when $T = 10$ the collaborative filtering performed at 1.11 whereas with the combined features the result is 1.05.

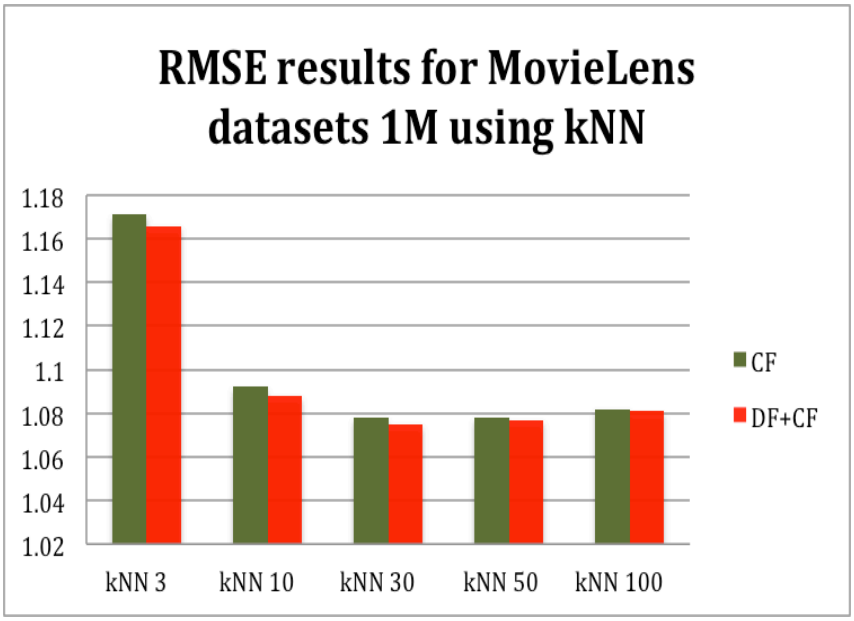

Figure 46: RMSE results for the MovieLens dataset using k-NN

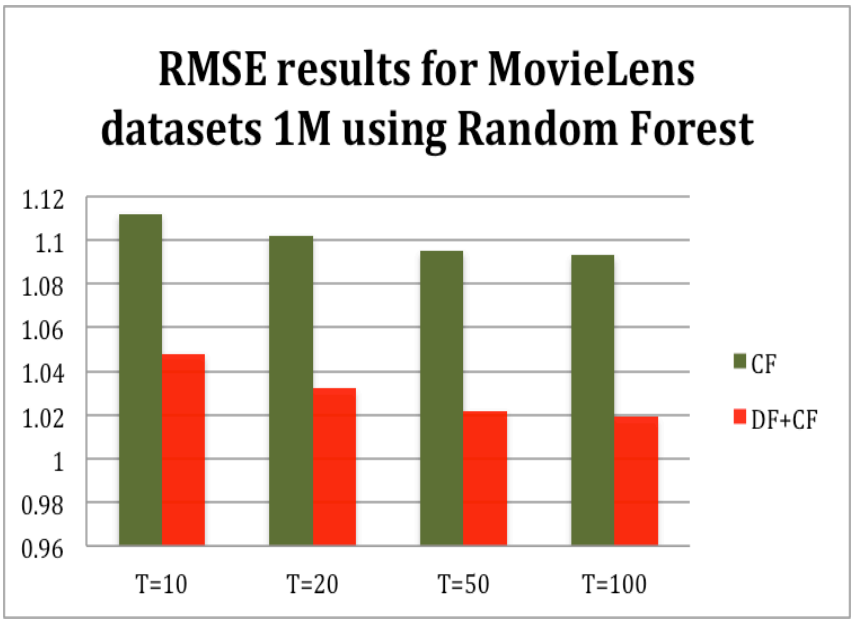

Figure 47: RMSE results for the MovieLens dataset using Random Forest

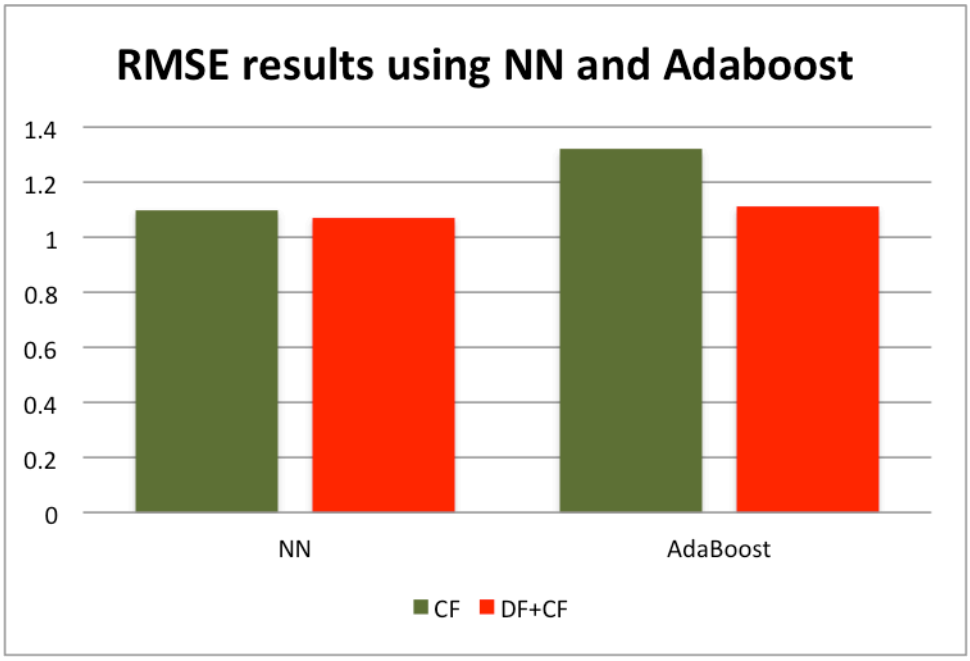

Figure 48: RMSE results for the MovieLens dataset using NN and AdaBoost

Figure 49 and Figure 50 show the improvement rate based on the results obtained from the MAE and RMSE experiments. It is shown that in both MAE and RMSE cases AdaBoost is the most improved method, followed by random forest, NN and kNN.

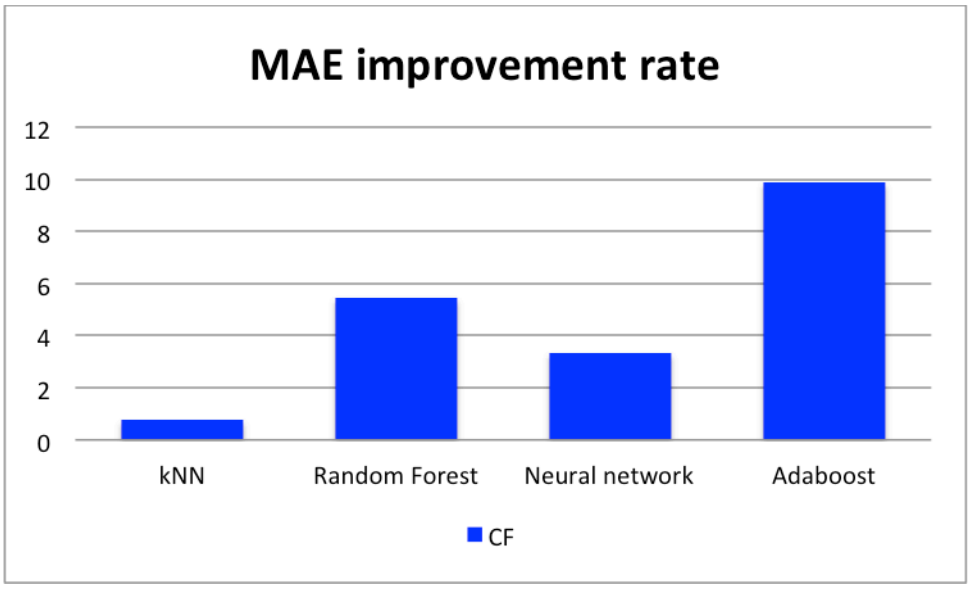

Figure 49: MAE improvement rate

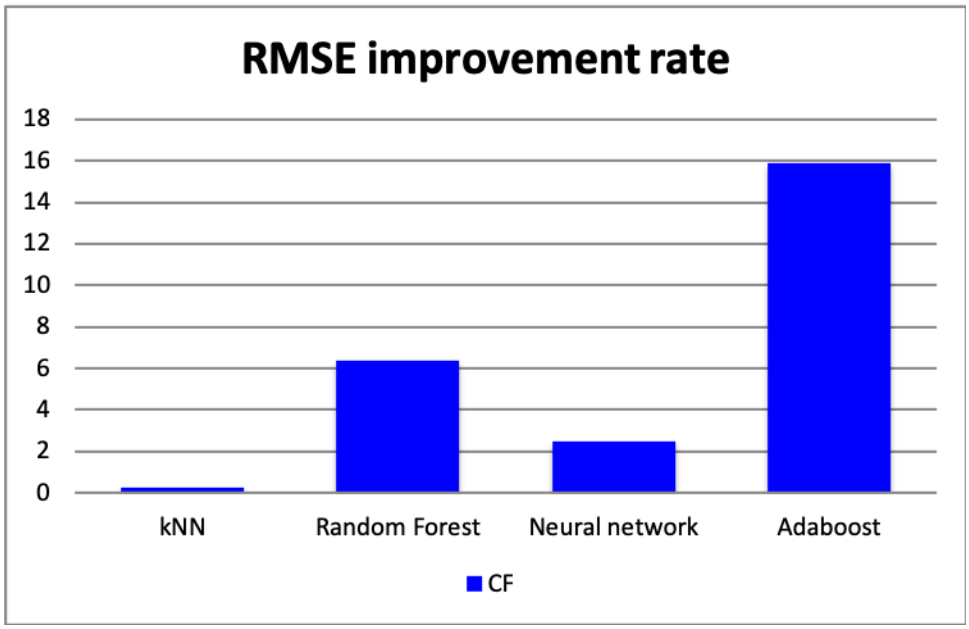

Figure 50: RMSE improvement rate

5.5.2 Discussion

Users rely on recommender systems to receive good recommendations in web environments, thus resulting to a reduced search burden for them and businesses rely to the possibility of better user experience and improved sales by utilizing recommendation technologies.

In this experiment a method that can compliment a business in their recommendation algorithm selection process is proposed. This particular method can be applied in domains where demographic information about user is available such as movie recommendation domains. Collaborative filtering combines with demographic information using a simple series of steps, thus resulting in improved recommendations for all evaluated classifiers. Such method is particularly useful for

a business in scenarios where a new method might be time consuming or costly to develop and an existing approach needs to be used.

By using the proposed method within a recommendation library and changing only the name of the recommendation algorithm and evaluating each it can be easily identified which one performs better for the given settings. For example when the kNN algorithm is applied there is a small but noticeable increase in terms of improvement, neural network is second with a higher improvement rate, random forest is third with an even higher improvement rate from neural network and AdaBoost was the one with the highest improvement rate. Thus, by noticing the MAE and RMSE improvement rates in Figure 49 and Figure 50 respectively. It can be easily identified that the prefered choice for a business would be AdaBoost.

5.6 Summary

This chapter has presented the results of all the proposed methods. It shows the significant value of the method compared to the baseline and the state-of-the-art algorithms. The experimental results showed that the proposed method significantly improved prediction accuracy and showed improvements when solving the long tail problem.

The chapter provided more detail on each experiment using various neighbourhoods, along with the experimental results, comparing the alternatives. Therefore, the experiment results indicate that switching hybrid method substantially outperforms collaborative filtering and content-based filtering in case of the items that belong to the tail. In addition, it evaluated the effectiveness of

using content-based as an alternative way that alleviates the long tail problem. Furthermore, the experiments demonstrated that the multi-level method contributes significant improvement with respect to different datasets compared to the traditional recommendation approaches.

6 Conclusions and future work

6.1 Conclusions

This chapter concludes the contribution of the thesis and discusses possible further work that might effectively be added on the top of this research.

This thesis presents a study of the most popular classification methods that have been used widely to generate the recommendations by reviewing each method with the advantages and the limitations from the literature. Therefore, this research found a gap in the knowledge that needs to be considered to improve the quality of the recommendation and avoid the challenges that are founded in the traditional measures and approaches.

Moreover, this thesis has presented a new hybrid approach that solves the long tail problem in recommender systems by applying a switching CBR and multi-level algorithm. First of all, the switching method was proposed with traditional similarity measures such as Pearson and Euclidean distance. When the system is asked for a new item, it obtains the number of ratings received for that movie. If the movie does not have a sufficient number of ratings then this movie faces the long tail problem. The system then switches to the CBR approach to calculate the rating. In this case, the CBR system uses movies that the user has rated in the past. Then, this method retrieves the majority of similar movies based on the description of the item in the user history for specific genres. Next, the rating of the new movie is calculated according to the most similar movie.

The advantage of the proposed method is that it does not need to pre-process the data before executing the recommendation. In addition, this method does not save

more information because of the fact that both techniques use the same information (user rating histories). Furthermore, the proposed method increases the accuracy of detecting and applying a CBR method with items in the long tail. Hence, this method solves the long tail recommendation problem and increase the accuracy of the recommendations. Moreover, it utilises the most used similarity measures to produce recommendations.

The switching method is further extended with the multi-level algorithm that solves the long tail recommendation problem and further improves the prediction accuracy. This method utilises multi-level algorithms to find similar users within a number of constraints. Those constraints provide a higher similarity value between users that have more common items, and the similarity value using the Pearson equation is above a defined threshold, which is something not addressed in the other methods. In addition, the co-rating items is considered in this method, which provides further insights into the benfits of enhancing the similarity as this is absent in the baseline CF method. Furthermore, this method improves the quality of prediction accuracy and solves the long tail recommendation problem.

In addition to that, a novel item-based collaborative filtering method based on triangle similarity and a multi-level algorithm was then proposed. This method was experimentally evaluated using three real datasets, along with a comparison to the traditional baseline Pearson correlation and to the state-of-the-art multi-level CF method. The results clearly show the quality of the recommendation is improved in all cases and that this system outperforms the alternatives.

Finally, a hybrid feature combination method was suggested based on a simultaneous combination of user-based collaborative filtering and demographic

features. The results suggest that demographic filtering can effectively improve the overall recommendation. Moreover, the proposed method addresses the common challenge of recommender systems, namely sparsity of data, through improving accuracy. Combining the hidden relations between users and comparing four different classifiers with a large dataset led to this improvement. This method also demonstrates the quality of the AdaBoost, Random Frost and Neural Network classifiers compared with k-NN when the demographic attributes are applied.

However, there could be further improvement to the contribution of this thesis and possibilities to extende the proposed methods, as explained in detail in the next section.

6.2 Future work

During studying and developing this recommender system, several ideas appeared that need to be researched further. This was due to the time limitation; this research focused on the research questions and objectives.

It is evident from the output of this thesis in terms of the best predictive accuracy results that was achieved through conducting a multi-level switching hybrid recommender system. However, in this thesis only one feature was employed to find the similarity of the items that users have rated in the past. It is possible that other content features could be investigated to find more precise items and increase the range of item relations. For example, the Internet Movie Database (IMDb) has many features that could be extracted to represent each item, features such as actor, actress, year and so on, which may produce more accurate recommendations. In this case, the genetic algorithm could also be adopted to compute a weight between two users using those features, since genetic algorithms have been combined with CBR

to find the optimal weight of related features. Hence, in recommender systems, genetic algorithms could be used to find the optimal similarity metrics.

Additionally, in this thesis, the focus was only on the rating prediction problem to evaluate the proposed methodology for recommendation algorithms. Therefore, the ranking task that considers the top-N recommendations list needs to be tested with the proposed method. Furthermore, the ranking task takes into account the explicit and implicit feedback to generate a relevant recommendation. Hence, other dimensions such as diversity and novelty become more important to evaluate the ranking algorithms in the context of recommendations.

Besides that, a user study and online evaluation bring a way of evaluating the quality of the recommendation algorithms from users perspectives. In addition, when users are involved in the recommendation system the reality of the study becomes visible and users can give some feedback to further improve the system. However, several limitations that prevent the researchers to evaluate the recommender systems from a user study. First, then it is very costly to conduct the experiment with real users especially with running more than one experiment since users are not willing to participate in all experiments. Second, a different population of users need to be considered in terms of age, gender, expertise and education to represent accurate evaluation. Lastly, user study can end up with biased results since users are aware that they are evaluating the system.

On the other hand, the deep learning algorithm plays an important role in several areas in recommender system, and to the best of this researcher's knowledge, it is not being exploited in regards to the long tail problem.

Due to the major challenges of recommender systems nowadays, such as information overload, deep learning modelling may help in alleviating the complexity of finding a user pattern to match with the closest similar user. The reason actually of chosen the deep learning as an extend work is that more recently, the leading conference in the recommender system community, RecSys, [1] established a workshop that focuses on the deep learning applications that are related to recommender systems. This workshop has gained a high volume of attention from researchers interested in developing new methods based on deep learning algorithms (S. Zhang, Yao & Sun, 2017).

Furthermore, deep learning is considered a sub field of machine learning techniques, that allow algorithms to learn from the past and understand concepts based on hierarchal terms. It is also a powerful method that considers multiple level problems and learns and adapts in order to optimise different tasks using deep learning representation. For example, content information can help the deep learning method to obtain missing rating values (H. Wang, Wang & Yeung, 2014).

Based on the literature surveys that more recently conducted a comprehensive systematic reviews for the recent papers related to deep learning recommender systems, several models are detailed in this chapter in order to provide a clear comparison between them.

Multilayer perceptron (MLP):

This is a feed forward neural network with multiple hidden layers between the input and the output. It has the ability to capture nonlinear transformation and find more

[1] https://recsys.acm.org/

relations between the user and the items. It can also conduct hierarchical learning to further investigate the feature representation.

Convolutional Neural Network (CNN):

This is also a feed forward neural network model but with convolution layers and pooling operations. It has the ability to capture local and global features that can help in improving the efficiency and the accuracy of the recommendation. It can help performing well the data as pre-processing step.

Recurrent Neural Network (RNN):

This method makes it more convenient to model a sequential data using loops, a long-term memory and a short-term memory. It is a strong candidate for the sequence pattern matching and identification due to its consecutive structure and parsing of prev-current-next sequence entries for a given domain. RNN is able to predict the next item in a given set of traces of previous ones (Mikolov, Karafiát, Burget, Černocký & Khudanpur, 2012).

Deep Reinforcement Learning (DRL):

DRL operates based on a trial and error paradigm. It is a combination between reinforcement learning and deep neural network that have attained very high level in terms of performance. Therfore, It has been used across a difficult task such as games and self driving cars (S. Zhang et al., 2017).

The strength of deep learning algorithms is that they are capable of modelling non-linear data, which makes it possible to address complexity and sophisticated user-

item interaction patterns. This is a challenge for matrix factorisation methods that utilise the linear function to combine user-item latent factors. In addition, deep learning is an efficient way to learn a representation of the available content that enables the algorithms to reduce the effort needed in relation to sparse data. It has the ability to combine different types of content such as text, image, audio and so on to produce as much helpful information as possible. Furthermore, a sequence model can learn based on sequential pattern modelling tasks such as natural language processing, speech recognition and so on. Finally, the flexibility of the available open source framework that has been implemented with the advent of many deep-learning algorithms makes it easily reachable.

On the other hand, there are some limitations that might affect the use of deep learning models in recommender systems. A common concern is about the hidden layers of the deep neural network.

To sum up, this research has presented a solid switching hybrid method that solved the long tail recommendation problem. this research also made the key idea of attention to utilise both collaborative filtering and content-based filtering that achieved more accurate results and solve the major challenges related to the items with an insufficient number of ratings. In addition, it pointed out the effectiveness of changing the similarity function and how it contributed to the recommender system model.

References

Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, *7*(1), 39–59. https://doi.org/10.1.1.56.4481

Adomavicius, G., Rokach, L., & Shapira, B. (2015). *Recommender Systems Handbook*. (F. Ricci, L. Rokach, & B. Shapira, Eds.), *Media* (Vol. 54). Boston, MA: Springer US. https://doi.org/10.1007/978-0-387-85820-3

Aggarwal, C. C. (2016). *Neighborhood-based collaborative filtering. Recommender Systems*. Springer. https://doi.org/10.1002/9781119054252

Ahn, H. J. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem, *178*, 37–51. https://doi.org/10.1016/j.ins.2007.07.024

Alshammari, G., Jorro-aragoneses, J. L., Kapetanakis, S., Petridis, M., Recio-garcía, J. A., & Diaz-Agudo, B. (2017). A hybrid CBR approach for the long tail problem in recommender systems. In *International Conference on Case-Based Reasoning* (pp. 35–45). Springer, Cham.

Alshammari, G., Jorro-aragoneses, J. L., Kapetanakis, S., Polatidis, N., & Petridis, M. (2019). A switching approach that improves prediction accuracy for long tail recommendations (pp. 1–12). Springer.

Alshammari, G., Kapetanakis, S., Polatidis, N., & Petridis, M. (2018). A Triangle Multi-Level Item-Based Collaborative Filtering Method That Improves Recommendations. In *International Conference on Engineering Applications of Neural Networks* (pp. 145–157). Springer.

Alshammari, G., Kapetanakis, S., Polatidis, N., Petridis, M., & Alshammari, A. (n.d.). A hybrid feature combination method that improves recommendations,

1–10.

Amatriain, X., Lathia, N., Pujol, J. M., Kwak, H., & Oliver, N. (2009). The wisdom of the few. *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '09*, 532. https://doi.org/10.1145/1571941.1572033

Amo, S. De, Diallo, M. S., Diop, C. T., Giacometti, A., Li, D., & Soulet, A. (2015). Contextual preference mining for user profile construction. *Information Systems*, *49*, 182–199. https://doi.org/10.1016/j.is.2014.11.009

Anand, S. S., Kearney, P., & Shapcott, M. (2007). Generating semantically enriched user profiles for Web personalization. *ACM Trans. Inter. Tech.*, *7*(4), 22-es. https://doi.org/10.1145/1278366.1278371

Anderson, C. (2007a). The Long Tail: Why the Future of Business Is Selling Less of More by Chris Anderson. *Journal of Product Innovation Management*, *24*(3), 1–30. https://doi.org/10.1111/j.1540-5885.2007.00250.x

Anderson, C. (2007b). The Long Tail: Why the Future of Business Is Selling Less of More by Chris Anderson. *Journal of Product Innovation Management*, *24*(3), 1–30. https://doi.org/10.1111/j.1540-5885.2007.00250.x

Armentano, M. G., Christensen, I., & Schiaffino, S. (2015). Applying the Technology Acceptance Model to Evaluation of Recommender Systems, (51), 73–79.

Bay, C., Kong, H., Building, S., & District, H. (2010). Collaborative Filtering Meets Mobile Recommendation : A User-centered Approach. *Science And Technology*, 236–241. Retrieved from http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/viewPDFInterstitial/1615/1964

Beel, J., & Langer, S. (2015). A Comparison of Offline Evaluations, Online Evaluations, and User Studies in the Context of Research-Paper Recommender Systems. In *Research and Advanced Technology for Digital Libraries* (pp. 153–168). Springer Science $\mathplus$ Business Media. https://doi.org/10.1007/978-3-319-24592-8_12

Bell, R. M., & Koren, Y. (2007). Improved neighborhood-based collaborative filtering. *KDD Cup and Workshop at the 13th ACM SIGKDD ...*, 7–14. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.443.300&rep=rep1&type=pdf%0Apapers3://publication/uuid/B4733BD0-54A5-4CAF-A7B2-6E851345BCB2

Berisha-boh, S. (2006). Defining User Profile to Improve Knowledge Extraction, *6*(7), 100–107.

Bhadoria, R. S., Sain, D., & Moriwal, R. (2011). Data Mining Algorithms for personalizing user ' s profiles on Web. *International Journal of Computer Technology and Electronics Engineering (IJCTEE)*, *1*(2), 120–125.

Bichindaritz, I. (2015). Data Mining Methods for Case-Based Reasoning in Health Sciences, 184–198.

Billsus Daniel, & Pazzani Michael J. (2002). User modeling for adaptative news access. . *User Modeling and User-Adapted Interaction*, *10*, 147–180. Retrieved from www.fxpal.com/people/billsus/pubs/um4news.pdf

Blanco-Fernández, Y., Pazos-Arias, J. J., Gil-Solla, A., Ramos-Cabrer, M., López-Nores, M., García-Duque, J., … Díaz-Redondo, R. P. (2008). Exploiting synergies between semantic reasoning and personalization strategies in intelligent recommender systems: A case study. *Journal of Systems and*

Software, *81*(12), 2371–2385. https://doi.org/10.1016/j.jss.2008.05.009

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, *46*. https://doi.org/10.1016/j.knosys.2013.03.012

Bogers, T., & Van Den Bosch, A. (2009). *Collaborative and content-based filtering for item recommendation on social bookmarking websites*. *CEUR Workshop Proceedings* (Vol. 532). https://doi.org/10.1007/978-0-387-85820-3

Breiman, L. (2001). RANDOM FORESTS, 1–33.

Bremer, S., Schelten, A., Lohmann, E., & Kleinsteuber, M. (2017). A Framework for Training Hybrid Recommender Systems, 30–37.

Bridge, D., Goker, M. H., McGinty, L., & Smyth, B. (2005). Case-based recommender systems. *The Knowledge Engineering Review*, *20*(03), 315. https://doi.org/10.1017/S0269888906000567

Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments 1. *User Modeling and User-Adapted Interaction*, 1–29.

Burke, R. (2007). Hybrid web recommender systems. *The Adaptive Web*, 377–408. https://doi.org/10.1007/978-3-540-72079-9_12

Callan, J., & Smeaton, A. (2003). Personalisation and Recommender Systems in Digital Libraries Joint NSF-EU DELOS Working Group Report. *Joint NSFEU DELOS Working Group Report Retrieved 1 January*, *5*(May), 299–308. https://doi.org/10.1007/s00799-004-0100-1

Cao, C., Ni, Q., & Zhai, Y. (2015). An improved collaborative filtering recommendation algorithm based on community detection in social networks. *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, (Iccsnt), 1–8. https://doi.org/10.1145/2739480.2754670

Celma, Ò. (2008). Music recommendation and discovery in the long tail. *Media*, *32*(3), 1–252. https://doi.org/10.1007/978-3-642-13287-2

Celma, Ò., & Herrera, P. (2008). A new approach to evaluating novel recommendations. In *Proceedings of the 2008 {ACM} conference on Recommender systems - {RecSys} {\textquotesingle}08*. Association for Computing Machinery ({ACM}). https://doi.org/10.1145/1454008.1454038

Chapelle, O., Vapnik, V., Bousquet, O., & Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, *46*(1–3), 131–159. https://doi.org/10.1023/A:1012450327387

Chen, Y., Wu, C., Xie, M., & Guo, X. (2011). Solving the sparsity problem in recommender systems using association retrieval. *Journal of Computers*, *6*(9), 1896–1902. https://doi.org/10.4304/jcp.6.9.1896-1902

Cotter, P., & Smyth, B. (2000). Ptv: Intelligent personalised tv guides. *Proceedings of the National Conference on Artificial Intelligence*, 957–964. Retrieved from http://www.aaai.org/Papers/IAAI/2000/IAAI00-004.pdf

Craw, S., Horsburgh, B., & Massie, S. (2015). Music recommendation: Audio neighbourhoods to discover music in the long tail. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9343*, 73–87. https://doi.org/10.1007/978-3-319-24586-7_6

Cremonesi, P., Koren, Y., & Turrin, R. (2010a). Performance of recommender algorithms on top-n recommendation tasks. *Proceedings of the Fourth ACM Conference on Recommender Systems - RecSys '10*, 39. https://doi.org/10.1145/1864708.1864721

Cremonesi, P., Koren, Y., & Turrin, R. (2010b). Performance of recommender

algorithms on top-n recommendation tasks. *Proceedings of the Fourth ACM Conference on Recommender Systems - RecSys '10*, 39. https://doi.org/10.1145/1864708.1864721

Cremonesi, P., Turrin, R., Lentini, E., & Matteucci, M. (2010). An evaluation methodology for recommender systems, (March).

Cursada, L. a, Carrera, D. E. S. U., & La, E. N. (2012). Novelty and Diversity Evaluation and Enhancement in Recommender Systems. *Recommender Systems Handbook*, (February), 5–8. Retrieved from http://dblp.uni-trier.de/db/reference/sp/rsh2015.html#CastellsHV15

Desrosiers, C., & Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. *In Recommender Systems Handbook Springer, Boston, MA.*, 107–144.

Duzen, Z., & Aktas, M. S. (2016). An approach to hybrid personalized recommender systems. *Proceedings of the 2016 International Symposium on INnovations in Intelligent SysTems and Applications, INISTA 2016*. https://doi.org/10.1109/INISTA.2016.7571865

Fathy, N. (2014). A Personalized Approach for Re-ranking Search Results Using User Preferences, *20*(9), 1232–1258.

Fleder, D., & Hosanagar, K. (2009). Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity. *Management Science*, *55*(5), 697–712. https://doi.org/10.1287/mnsc.1080.0974

Fleeson, W., Jayawickreme, E., Jones, A. B. A. P., Brown, N. A., Serfass, D. G., Sherman, R. A., … Matyjek-, M. (2017). The Use of the Genetic Algorithms in the Recommender Systems. *Journal of Personality and Social Psychology*, *1*(1), 1188–1197. https://doi.org/10.1111/j.1469-7610.2010.02280.x

Garcin, F., & Faltings, B. (2013). Pen recsys: a personalized news recommender systems framework. In *2013 International News Recommender Systems Workshop and Challenge*. https://doi.org/10.1145/2516641.2516642

Gauch, S., Gauch, S., Speretta, M., Speretta, M., Chandramouli, A., Chandramouli, A., … Micarelli, A. (2007). User Profiles for Personalized Information Access. *The Adaptive Web*, *4321*, 54–89. https://doi.org/10.1007/978-3-540-72079-9_2

Gedikli, F., & Jannach, D. (2010). Recommending based on rating frequencies : Accurate enough ? *In Proceedings of the 8th Workshop on Intelligent Techniques for Web Personalization & Recommender Systems at UMAP'10 (ITWP'10)*, 65–70.

Gedikli, F., Jannach, D., & Ge, M. (2014). How should I explain? A comparison of different explanation types for recommender systems.

George, T. (2005). A Scalable Collaborative Filtering Framework based on Co-clustering. *Data Mining 5th IEEE Conf*.

Ghazanfar, M., & Prugel-Bennett, A. (2010). An Improved Switching Hybrid Recommender System Using Naive Bayes Classifier and Collaborative Filtering. *Computer*, *I*, 493–502. Retrieved from http://eprints.ecs.soton.ac.uk/18483/

Gipp, B., Beel, J., & Hentschel, C. (2009). Scienstein : A Research Paper Recommender System. *Scienstein : A Research Paper Recommender System*, (January 2009), 309–315. https://doi.org/10.1109/WI.2006.149

Glauber, R., Loula, A., & Rocha-Junior, J. B. (2013). A mixed hybrid recommender system for given names. *CEUR Workshop Proceedings*, *1120*, 25–36.

Godoy, D., & Amandi, A. (2005). User profiling in personal information agents: a survey. *The Knowledge Engineering Review*, *20*(04), 329.

https://doi.org/10.1017/S0269888906000397

Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the {ACM}*, *35*(12), 61–70. https://doi.org/10.1145/138859.138867

Gunawardana, A., & Shani, G. (2009). A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. *The Journal of Machine Learning Research*, *10*, 2935–2962. https://doi.org/10.1145/1577069.1755883

Gupta, J. (2015). Performance Analysis of Recommendation System Based On Collaborative Filtering and Demographics.

Hannak, A., Sapiezynski, P., Kakhki, A., Krishnamurthy, B., Lazer, D., Mislove, A., & Wilson, C. (2013). Measuring personalization of web search. ... *on World Wide Web*, 527–537. Retrieved from http://dl.acm.org/citation.cfm?id=2488435

Hannon, J., Bennett, M., & Smyth, B. (2010). Recommending Twitter Users to Follow Using Content and Collaborative Filtering Approaches. *Search*, *Barcelona,*(9), 199–206. https://doi.org/10.1145/1864708.1864746

Hannon, J., McCarthy, K., & Smyth, B. (2011). Finding Useful Users on Twitter: Twittomender the Followee Recommender. *Science*, *6611*(ADVANCES IN INFORMATION RETRIEVAL), 784–787. https://doi.org/10.1007/978-3-642-20161-5_94

Harper, F. M., & Konstan, J. A. (2015a). The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.*, *5*(4), 19:1----19:19. https://doi.org/10.1145/2827872

Harper, F. M., & Konstan, J. A. (2015b). The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.*, *5*(4), 19:1--19:19.

https://doi.org/10.1145/2827872

Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international {ACM} {SIGIR} conference on Research and development in information retrieval - {SIGIR} {\textquotesingle}99*. Association for Computing Machinery ({ACM}). https://doi.org/10.1145/312624.312682

Herlocker, J. L., Konstan, J. a., Terveen, L. G., & Riedl, J. T. (2004a). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, *22*(1), 5–53. https://doi.org/10.1145/963770.963772

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004b). Evaluating collaborative filtering recommender systems. *{ACM} Transactions on Information Systems*, *22*(1), 5–53. https://doi.org/10.1145/963770.963772

Hervas-drane, A. (2008). Word of Mouth and Recommender Systems : A Theory of the Long Tail. *Business*, 1–48.

Huang, M. J., Huang, H. S., & Chen, M. Y. (2007). Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach. *Expert Systems with Applications*, *33*(3), 551–564. https://doi.org/10.1016/j.eswa.2006.05.019

Im, K. H., & Park, S. C. (2007). Case-based reasoning and neural network based expert system for personalization. *Expert Systems with Applications*, *32*(1), 77–85. https://doi.org/10.1016/j.eswa.2005.11.020

Jannach, D., Lerche, L., Gedikli, F., & Bonnin, G. (2013). What Recommenders Recommend – An Analysis of Accuracy , Popularity , and Sales A Multi-metric Experimental Evaluation. *21th International Conference, UMAP 2013,*

Rome, Italy, June 10-14, 2013, 25–37. https://doi.org/10.1007/978-3-642-38844-6_3

Jeong, B., Lee, J., & Cho, H. (2010). Improving memory-based collaborative filtering via similarity updating and prediction modulation. *Information Sciences*, *180*(5), 602–612. https://doi.org/10.1016/j.ins.2009.10.016

Jeyshirii. (2014). Precise Recommendation System for the Long Tail Problem Using Adaptive Clustering, 3963–3974.

Johnson, J., & Ng, Y.-K. (2017). Enhancing long tail item recommendations using tripartite graphs and Markov process. *Proceedings of the International Conference on Web Intelligence - WI '17*, 761–768. https://doi.org/10.1145/3106426.3106439

Juan A. Recio-García Pedro A. González-Calero, B. D.-A. (2014). jcolibri2: A framework for building Case-based reasoning systems. *Science of Computer Programming*, *79*, 126–145. https://doi.org/10.1016/j.scico.2012.04.002

Kim, H., & Chan, P. (2006). Personalized Search Results with User Interest Hierarchies Learnt from Bookmarks. *Advances in Web Mining and Web Usage Analysis*, *4198*, 158–176. https://doi.org/10.1007/11891321_9

Kim, W., & Solutions, C. D. (2002). Personalization : Definition , Status , and Challenges Ahead, *1*(1), 29–40.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, *2*(0), 1137–1143. https://doi.org/10.1067/mod.2000.109031

Kohavi, Ron, Longbotham, R., Sommerfield, D., & Henne, R. M. (2008). Controlled experiments on the web: survey and practical guide. *Data Mining*

and Knowledge Discovery, *18*(1), 140–181. https://doi.org/10.1007/s10618-008-0114-1

Kolodner, J. L. (1992). An introduction to case-based reasoning. *Artificial Intelligence Review*, *6*(1), 3–34. https://doi.org/10.1007/BF00155578

Konstan, J. A., & Riedl, J. (2012). Recommender systems: From algorithms to user experience. *User Modeling and User-Adapted Interaction*, *22*(1–2), 101–123. https://doi.org/10.1007/s11257-011-9112-x

Koren, Y. (2008). Factorization meets the neighborhood: a Multifaceted Collaborative Filtering Model. *Kdd 08*, 9. https://doi.org/10.1145/1401890.1401944

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer*, *42*(8), 30–37. https://doi.org/10.1109/mc.2009.263

Kumar, Rajeev. (2014). Social Popularity based SVD ++ Recommender System, *87*(14), 33–37.

Kumar, Rakesh, & Sharan, A. (2014). Personalized web search using browsing history and domain knowledge. *2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*, 493–497. https://doi.org/10.1109/ICICICT.2014.6781332

Lam, X., Vu, T., & Le, T. (2008). Addressing cold-start problem in recommendation systems. *Proceedings of the 2nd International*, 208–211. https://doi.org/10.1145/1352793.1352837

Laveti, R. N., Ch, J., Pal, S. N., & Babu, N. S. C. (2016). 2016 IEEE 23rd International Conference on High Performance Computing Workshops A Hybrid Recommender System using Weighted Ensemble Similarity Metrics

and Digital Filters. https://doi.org/10.1109/HiPCW.2016.14

Li, F., Sun, J., & Zhangi, X. (2015). Study on the Key Technology of Personalized Recommendation of Case-based Reasoning, *8*(4), 377–382.

Li, L., Yang, Z., Wang, B., & Kitsuregawa, M. (2007). Dynamic adaptation strategies for long-term and short-term user profile to personalize search. *Joint 9th Asia-Pacific Web Conference, APWeb 2007, and 8th International Conference, on Web-Age Information Management, WAIM 2007, Advances in Data and Web Management*, *4505*, 228–240. https://doi.org/10.1007/978-3-540-72524-4_26

Linden, G., Smith, B., & York, J. (2003). Amazon.com recommendations: item-to-item collaborative filtering. *{IEEE} Internet Computing*, *7*(1), 76–80. https://doi.org/10.1109/mic.2003.1167344

Liu, F., Yu, C., & Meng, W. (2004). Personalized Web Search for Improving Retrieval Effectiveness. *IEEE Transactions on Knowledge and Data Engineering*, *16*(1), 28–40. https://doi.org/10.1109/TKDE.2004.1264820

Louppe, G., Wehenkel, L., Sutera, A., & Geurts, P. (2013). Understanding variable importances in forests of randomized trees. *Neural Information Processing Systems*, 1–9. https://doi.org/NIPS2013_4928

Lu, J., Wu, D., Mao, M., Wang, W., & Zhang, G. (2015). Recommender System Application Developments: A Survey. *Decision Support Systems*, *74*, 12–32. https://doi.org/10.1016/j.dss.2015.03.008

Lü, L., Medo, M., Yeung, C. H., Zhang, Y.-C., Zhang, Z.-K., & Zhou, T. (2012a). Recommender systems. *Physics Reports*, *519*(1), 1–49. https://doi.org/10.1016/j.physrep.2012.02.006

Lü, L., Medo, M., Yeung, C. H., Zhang, Y.-C., Zhang, Z.-K., & Zhou, T. (2012b).

Recommender systems. *Physics Reports*, *519*(1), 1–49. https://doi.org/10.1016/j.physrep.2012.02.006

Ma, H., King, I., & Lyu, M. R. (2007). Effective missing data prediction for collaborative filtering. *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '07*, 39. https://doi.org/10.1145/1277741.1277751

Malone, T. W. (1986). Intelligent information sharing systems. *Management Research News*, *15*(8), 1–5. https://doi.org/10.1108/eb028253

Martin-vicente, M., Gil-Solla, A., Ramos-Cabrer, M., Blanco-Fernandez, Y., & Lopez-Nores, M. (2010). A semantic approach to avoiding fake neighborhoods in collaborative recommendation of coupons through digital {TV}. *{IEEE} Transactions on Consumer Electronics*, *56*(1), 54–62. https://doi.org/10.1109/tce.2010.5439126

MelvilleP, M. R. J., & R, N. (2002). Content boosted collaborative filtering for improved recommendations. *Proceedings of the 18th National Conference on Artificial Intelligence*, 187–-192.

Mikolov, T., Karafiát, M., Burget, L., Černocký, J., & Khudanpur, S. (2012). Recurrent neural network based language model, (5), 89. https://doi.org/10.1109/ICWS.2017.130

Mittal, P. (2014). Metadata Based Recommender Systems, 2659–2664.

Moawad, I. F., Talha, H., Hosny, E., & Hashim, M. (2012). Agent-based web search personalization approach using dynamic user profile. *Egyptian Informatics Journal*, *13*(3), 191–198. https://doi.org/10.1016/j.eij.2012.09.002

Mobasher, B. (2007). Data Mining for Web Personalization. *Adaptive Web*, 90–135.

Moldovan, A. N., & Muntean, C. H. (2009). Personalisation of the multimedia

content delivered to mobile device users. *2009 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, BMSB 2009*. https://doi.org/10.1109/ISBMSB.2009.5133750

Nanda, A., Omanwar, R., & Deshpande, B. (2014). Implicitly Learning a User Interest Profile for Personalization of Web Search Using Collaborative Filtering. *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, *2*, 54–62. https://doi.org/10.1109/WI-IAT.2014.80

Nicolas, H. (2017). {S}urprise, a {P}ython library for recommender systems. Retrieved from http://surpriselib.com

Park, Y.-J., & Tuzhilin, A. (2008). The long tail of recommender systems and how to leverage it. *Proceedings of the 2008 ACM Conference on Recommender Systems RecSys 08*, 11. https://doi.org/10.1145/1454008.1454012

Park, Y. J. (2013). The adaptive clustering method for the long tail problem of recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, *25*(8), 1904–1915. https://doi.org/10.1109/TKDE.2012.119

Pazzani, M. J. (1999). A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, *13*(5), 393–408. https://doi.org/10.1023/A:1006544522159

Pazzani, M. J., & Billsus, D. (2007). Content-based recommendation systems. *The Adaptive Web*, 325–341. Retrieved from http://dl.acm.org/citation.cfm?id=1768197.1768209

Pierrakos, D., Paliouras, G., & Ioannidis, Y. (2012). OurDMOZ : A System for Personalizing the Web . *6th International Workshop on Personalized Access Profile Management, and Context Awareness in Databases.*

Polatidis, N., & Georgiadis, C. K. (2016). A multi-level collaborative filtering method that improves recommendations, *48*, 100–110.

Pradesh, U. (2018). Recommendation research trends : review , approaches and open issues Anu Taneja * and Anuja Arora, *13*(2), 123–186.

Qian, X., Feng, H., Zhao, G., & Mei, T. (2014). Personalized recommendation combining user interest and social circle. *IEEE Transactions on Knowledge and Data Engineering*, *26*(7), 1763–1777. https://doi.org/10.1109/TKDE.2013.168

Redpath, J. L. (2010). Improving the Performance of Recommender Algorithms.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens : An Open Architecture for Collaborative Filtering of Netnews. *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, 175–186. https://doi.org/10.1145/192844.192905

Resnick, P., & Varian, H. R. (1997). Recommender systems. *Communications of the {ACM}*, *40*(3), 56–58. https://doi.org/10.1145/245108.245121

Rousseau, B., Browne, P., Malone, P., & Ó Foghlú, M. (2004). User profiling for content personalisation in information retrieval. *ACM Symposium on Applied Computing (SAC 2004)*.

Sarwar, B. M., Karypis, G., Konstan, J., & Riedl, J. (2002). Recommender Systems for Large-scale E-Commerce : Scalable Neighborhood Formation Using Clustering. *In Proceedings of the Fifth International Conference on Computer and Information Technology*, *1*, 291–324.

Se, U., Haracteristics, C., Mpact, I., Xiao, B., & Benbasat, I. (2007). E-Commerce P Roduct R Ecommendation a Gents :Use, Characteristics, and Impact1. *MIS Quarterly*, *31*(1), 137–209. https://doi.org/10.2307/25148784

Semeraro G., L. P. B. P. G. M. d. (2009). Knowledge Infusion into Content-based Recommender Systems. *Proceedings of the 2009 ACM Conference on Recommender Systems*, 22–25.

Shani, G., Brafman, R. I., & Shimony, S. E. (2005). Model-Based Online Learning of {POMDPs}. In *Machine Learning: {ECML} 2005* (pp. 353–364). Springer Science \mathplus Business Media. https://doi.org/10.1007/11564096_35

Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. *Recommender Systems Handbook*, 257–298. https://doi.org/10.1007/978-0-387-85820-3_8

Sharma, N., Sharma, M., & Gupta, O. J. (2012). Search Engine Personalization Using Concept Based User Profiles, *1*(July), 84–87.

Shen, K., Liu, Y., & Zhang, Z. (2017). Modified Similarity Algorithm for Collaborative Filtering, 378–385. https://doi.org/10.1007/978-3-319-62698-7

Shi, Y. U. E., Larson, M., & Hanjalic, A. (2014). Collaborative Filtering beyond the User-Item Matrix : A Survey of the State of the Art and Future Challenges, *47*(1), 1–45.

Smyth, B. (2007). Case-Based Recommendation, 342–376.

Spiegel, S. (2009). A Hybrid Approach to Recommender Systems based on Matrix Factorization presented by. *Matrix*.

Steck, H. (2013). Evaluation of recommendations: rating-prediction and ranking, 213–220.

Sun, J., Zhao, Q., Antony, S., & Chen, S. (2015). Personalized Recommendation Systems : An Application in Case-based Reasoning, (Cas), 369–371.

Sun, S., Zhang, Z., Dong, X., Zhang, H., Li, T., Zhang, L., & Min, F. (2017). Integrating Triangle and Jaccard similarities for recommendation, 1–16.

Sun, T., Wang, L., & Guo, Q. (2009). A collaborative filtering recommendation algorithm based on Item similarity of user preference. *Proceedings - 2009 2nd International Workshop on Knowledge Discovery and Data Mining, WKKD 2009*, 60–63. https://doi.org/10.1109/WKDD.2009.90

Tan, Z., & He, L. (2017). An Efficient Similarity Measure for User-Based Collaborative Filtering Recommender Systems Inspired by the Physical Resonance Principle, *5*, 27211–27228.

Tandberg, S. S., & Øystein, L. (2015). Design of a Hybrid Recommender System : A Study of the Cold-Start User Problem. *Master's Thesis, NTNU*, (May).

Thorat, P. B., Goudar, R. M., & Barve, S. (2015). Survey on Collaborative Filtering , Content-based Filtering and Hybrid Recommendation System. *International Journal of Computer Applications*, *110*(4), 31–36.

Tintarev, N. (2009). Explaining recommendations. *User Modeling 2007*, 470–474.

Tintarev, Nava, & Masthoff, J. (2011). *Recommender Systems Handbook*. *Recommender Systems Handbook* (Vol. 54). https://doi.org/10.1007/978-0-387-85820-3

Vozalis, M., & Margaritis, K. G. (2004). Collaborative filtering enhanced by demographic correlation. *In AIAI Symposium on Professional Practice in AI, of the 18th World Computer Congress.*

Wang, H., Wang, N., & Yeung, D.-Y. (2014). Collaborative Deep Learning for Recommender Systems. https://doi.org/10.1145/2783258.2783273

Wang, Q., & Jin, H. (2010). Exploring online social activities for adaptive search personalization. *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, 999–1008. https://doi.org/http://doi.acm.org/10.1145/1871437.1871564

Wang, Shanfeng, Gong, M., Li, H., & Yang, J. (2016). Knowle dge-Base d Systems Multi-objective optimization for long tail recommendation, *104*, 145–155.

Wang, Shenghui, Computer, O., Stash, N., Eindhoven, T. U., Aroyo, L., Amsterdam, V. U., … Amsterdam, V. U. (2014). Enhancing Content-Based Recommendation with the Task Model of Classification Conference, *214*(October). https://doi.org/10.1007/978-3-642-54930-4

Wanvimol Nadee. (2016). MODELLING USER PROFILES FOR RECOMMENDER SYSTEMS Wanvimol Nadee.

Wasid, M., & Ali, R. (2018). An Improved Recommender System based on Multi-criteria Clustering Approach. *Procedia Computer Science*, *131*, 93–101. https://doi.org/10.1016/j.procs.2018.04.190

Weng, L. T., Xu, Y., Li, Y., & Nayak, R. (2007). Improving recommendation novelty based on topic taxonomy. *Proceedings - 2007 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT Workshops 2007*, 115–118. https://doi.org/10.1109/WIIATW.2007.4427553

Yin, H., Cui, B., Li, J., Yao, J., & Chen, C. (2012). Challenging the long tail recommendation. *Proceedings of the VLDB Endowment*, *5*(9), 896–907. https://doi.org/10.14778/2311906.2311916

Zhang, H., Min, F., & Wang, S. (2014). A Random Forest Approach to Model-based Recommendation ⋆, *15*(61379089), 5341–5348. https://doi.org/10.12733/jics20104701

Zhang, L., Tao, Q., & Teng, P. (2014). An Improved Collaborative Filtering Algorithm Based on User Interest. *Journal of Software*, *9*(Iccmcee), 999–1006. https://doi.org/10.4304/jsw.9.4.999-1006

Zhang, S., Yao, L., & Sun, A. (2017). Deep Learning based Recommender System: A Survey and New Perspectives, *1*(1), 1–35. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Zhou, J., & Luo, T. (2010). A novel approach to solve the sparsity problem in collaborative filtering. *2010 International Conference on Networking, Sensing and Control, ICNSC 2010*, 165–170. https://doi.org/10.1109/ICNSC.2010.5461512

Ziegler, C.-N. C. N., McNee, S. M. S. M., Konstan, J. a. J. a., & Lausen, G. (2005). Improving recommendation lists through topic diversification. *Proceedings of the 14th International Conference on World Wide Web WWW 05*, 22. https://doi.org/10.1145/1060745.1060754

Ziegler, C.-N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005). Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web - {WWW} {\textquotesingle}05*. Association for Computing Machinery ({ACM}). https://doi.org/10.1145/1060745.1060754