

# **Model-Based Learning to Augment Collaborative Filtering: Prediction and Evaluation**

A Dissertation

Presented in Partial Fulfillment of the Requirements for the  
Degree of Doctor of Philosophy

with a

Major in Computer Science

in the

College of Graduate Studies

University of Idaho

by

Ashrf Althbiti

Major Professor: Xiaogang Ma , Ph.D.


Committee Members: Terry Soule, Ph.D.; Stephen Lee, Ph.D.; Jia Song, Ph.D.

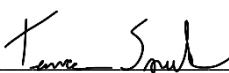
Department Administrator: Terry Soule, Ph.D.

May 2021

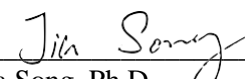
### Authorization to Submit Dissertation


This dissertation of Ashrf Althbiti, submitted for the degree of Doctor of Philosophy with a Major in Computer Science and titled, "Model-Based Learning to Augment Collaborative Filtering: Prediction and Evaluation," has been reviewed in final form. Permission, as indicated by the signatures and dates below, is now granted to submit final copies to the College of Graduate Studies for approval.

Major Professor:  Date: 04/06/2021  
Xiaogang Ma, Ph.D.

Committee Members:  Date: April 12, 2021  
Terry Soule, Ph.D.

 Date: 04/08/2021  
Stephen Lee, Ph.D.

 Date: April 8, 2021  
Jia Song, Ph.D.

Department Administrator:  Date: April 12, 2021  
Terry Soule, Ph.D.

## Abstract

Collaborative filtering (CF) is a novel statistical technique developed to retrieve useful information and to generate predictions based on provided data from users. It is fundamentally characterized by recommender systems (RSs), which have recently attracted researchers' attention. CF can be defined as systems and software tools that automatically and effectively generate a list of recommendations of the most suitable items to a target user by predicting a user's future ratings for unseen items. As a field of study, the dramatic evolution of machine learning solves problems that appeared in the early time of CF systems. Researchers claim that the main research focus of current CF research, especially in the big data era, is how to effectively develop models to address the problems of data sparsity and limited coverage that CF systems unexpectedly experience.

The particular objectives of this empirical research are: (1) providing a novel model based on machine learning and data mining algorithms that address the data sparsity problem in CF, (2) providing a novel similarity model based on rating alignment that results in better accuracy of rating prediction compared to the other similarity models used for CF, (3) providing a novel model to incorporate information from social network sites (SNSs) to augment CF and solve the data sparsity problem, and (4) providing academic advisory RS based on a web-based framework.

To validate the effectiveness of the proposed models, intensive experiments are conducted to compare the performance of the proposed models with the state-of-the-art CF models using four datasets collected from four popular RSs' domains (music, jokes, books, and movies). These proposed models are computationally efficient and effectively generalized to other related fields in RSs. Results retained from evaluation metrics reveal that the proposed models can demonstrate promising prediction accuracy and improve the prediction performance better than the state-of-the-art algorithms. Furthermore, the proposed models can successfully address the data sparsity and the limited coverage problems.

## Acknowledgements

First and foremost, I must praise and thank my Almighty God (Allah) for his blessings; without his grace and compassion, I would have nothing. He alone helps me to proceed and succeed in my life.

Then, I must acknowledge my sponsors, the government of Saudi Arabia and Taif University for supporting me to pursue the Ph.D. degree. Also, I must admit my appreciations to the University of Idaho, the College of Graduate Studies and the Department of Computer Science for allowing me to join their graduate program and conduct my research. Also, I convey my sincere appreciation to the chair and the management staff of the Department of Computer Science for the grant that they provided me.

Next, I must express my sincere appreciation to my respected major advisor Prof. Xiaogang Ma, for his invaluable support of my Ph.D. study and related research and for his patience, motivation, and immense knowledge. His supervision helps me at all the times while conducting research and writing of this dissertation. I could not have imagined conducting and finishing this research without Prof. Ma's help and guidance. Without his guidance, I would not have been able to work on such a cutting-edge topic and application of recommender systems. I am forever indebted to Prof. Ma's for supporting me over the whole duration of my Ph.D. work, not only in my academic work but also in all my personal matters. I have to thank him for the grants that he provided to participate in several conferences. I do not only get my knowledge from the courses that I have taken, but further I do gain enormous knowledge from Prof. Ma. I am proud to be one of Prof. Ma's students.

Moreover, I would like to convey my acknowledgements and appreciation from the bottom of my heart to the rest of my dissertation committee members for their insightful comments and encouragement throughout my Ph.D. work: Prof. Terry Soule, Prof. Stephen Lee, and Prof. Jia Song. I can't adequately express my sincere acknowledgements and appreciation to my committee members for their patience over the last two years. I am deeply indebted to all of them for providing much invaluable insight, support, and feedback throughout this process.

Besides my major advisor and committee members, I would like to recognize the members of IDEA lab who have all contributed to the progress I have made.

## Dedication

This work is dedicated to my parents, Wasllah Althubiati and Azizah Althubiati. I am not able to express my appreciation to them for all of the sacrifices that they have made on my behalf. Their invaluable unconditional and continuous love, care, and support since the first day of my life back in 1988 till where I am today and for the rest of my life is the real blessing that cannot be perceived by all other blessings. Without their prayers, no happiness could touch my heart.

Also, it is dedicated to my beloved family members including my brothers, my sisters, and my relatives. A special dedication goes to my beloved brother “Saud”. Without his care, love, logistical and financial support, I would not be able to proceed towards my goals. Also, this work is dedicated to my brother “Abdul Moaen”; I am forever thankful to you.

This work is particularly dedicated to my dear wife, Norah, who has spent sleepless nights taking care of our children while she pursuing a Ph.D. degree in the department of Physics at University of Idaho. You have been a constant source of support and encouragement during the challenges of graduate school and difficulties of life. I am blessed to have you in my life. Your encouragements are the fuel that drives me straight to achieve my goals.

I would like also to dedicate this work to my beloved children Abdullah and Deem, thank you for being in my life. Without the encouragements and the motivations of my family, I would not be able to achieve my goals.

Besides my family members, I would like to dedicate this work to the Dean of the Turabah University College: Prof. Saad bin Hamoud Al-Osaimi. I cannot thank you enough for all that you have done for me. This work is also dedicated to my best friend (Talal) who passed away back in 2013, and to everyone who belongs to Alotaibi’s tribe.

(خلال سنوات غربتي في طلب العلم، وجدت الفخر في ثلاثة أمور فقط، الأول فخراً بإسلامي، ثم فخراً بوطني وهويتي، والثالث فخراً

بأنني من عتيبة)

اشرف بن وصل الله الثبيتي

## Table of Contents

<b>Authorization to Submit Dissertation.....</b>	<b>ii</b>
<b>Abstract .....</b>	<b>iii</b>
<b>Acknowledgements .....</b>	<b>iv</b>
<b>Dedication.....</b>	<b>v</b>
<b>List of Tables.....</b>	<b>xi</b>
<b>List of Figures .....</b>	<b>xii</b>
<b>Chapter 1: Introduction.....</b>	<b>1</b>
Introduction .....	1
Problem Statement and Research Motivation .....	2
Scientific Contribution, Significance, and Impact.....	4
Research Objectives and Questions.....	6
Thesis Statement.....	7
List of Publications.....	7
Author’s Related Publications .....	7
Author’s Collaborative Papers not directly related to this Dissertation .....	7
Presentations in Conferences.....	8
Grants .....	8
Organization .....	8
<b>Chapter 2: A Literature Review on Model-Based Collaborative Filtering.....</b>	<b>11</b>
Introduction .....	11
The Potential Needs of CF .....	12
Types of Collected Data .....	12
CF Applications.....	14
CF Techniques.....	14
Neighborhood-based CF.....	14
User-based CF .....	15

Item-based CF .....	17
Model-based CF .....	17
Factorization Technique .....	18
Solving the Data Sparsity Problem Using Advanced Models .....	20
Graph-based CF.....	22
Hybrid-based CF .....	22
Evaluation Methods.....	24
Evaluation Settings.....	24
Evaluation Properties.....	26
Accuracy.....	26
Conclusion.....	29
<b>Chapter 3: State-of-the-Art Machine Learning Algorithms Applied to Collaborative Filtering</b>	<b>30</b>
Introduction .....	30
Data Representation and Feature Engineering .....	31
Machine Learning Algorithms .....	31
Classification and Regression.....	32
Linear regression .....	34
K-nearest neighbors (KNNs).....	34
Decision Trees (DSs).....	34
Bayesian classifiers .....	35
Support Vector Machines (SVMs) .....	35
Artificial Neural Networks (ANNs) .....	35
Dimensionality reduction .....	35
Clustering .....	35
Association rule mining.....	36
Model Fitness and Evaluation .....	36
Conclusion.....	36

<b>Chapter 4: Addressing Data Sparsity in Collaborative Filtering Based Recommender Systems</b>	
<b>Using Clustering and Artificial Neural Network.....</b>	<b>38</b>
Introduction .....	38
Related Research .....	39
The Proposed Model .....	40
Data Preprocessing .....	40
Clustering .....	43
Artificial Neural Network.....	43
Experiments.....	44
Datasets .....	44
Experiments Setup.....	46
Evaluation.....	46
Results .....	47
Discussion and Results Analysis .....	53
Conclusion.....	55
<b>Chapter 5: An Efficient Similarity Measure Based on Rating Alignment to Augment</b>	
<b>Collaborative Filtering.....</b>	<b>57</b>
Introduction .....	57
Related Research .....	58
Collaborative Filtering Techniques .....	59
Neighborhood-Based CF.....	59
Conventional Similarity Measures .....	61
Cosine Similarity .....	61
Pearson Correlation Coefficient .....	61
Mean Square Difference.....	62
Jaccard.....	62
Jaccard Mean Square Difference.....	63
Limitations of conventional similarity measures.....	63



Proposed Similarity Model.....	63
The Motivation of the Proposed Model.....	63
Proposed Rating Alignment-Based Similarity Model .....	64
Rating Prediction.....	65
Time Complexity.....	65
Experiments.....	65
Datasets .....	66
Evaluation Metrics.....	68
Experiments Setup.....	68
Results of the Experiments.....	69
Discussion .....	75
Conclusion.....	77
<b>Chapter 6: A Personalized Academic Advisory Recommender System.....</b>	<b>79</b>
Introduction .....	79
Literature Survey.....	80
Approach .....	82
Content-Based Filtering.....	85
k-Nearest Neighbors (kNN) .....	88
PAARS Implementation.....	89
Tools and Technical Component.....	89
Data Preprocessing .....	91
PAARS Web-Based Application.....	91
Experiments and Evaluation.....	92
Conclusion.....	93
<b>Chapter 7: Fusing Social Network Influencers' Data to Augment Recommender Systems.....</b>	<b>95</b>
Introduction .....	95
Literature Review .....	96

Social Network Sites .....	96
Recommender Systems .....	96
Bayesian Classifier .....	97
Proposed Approach .....	97
Conclusion.....	99
<b>Chapter 8: Conclusions, Limitations, and Future Research Directions.....</b>	<b>100</b>
Introduction .....	100
Summary .....	100
Limitations.....	102
Future Work .....	102
Extending the Presented Models .....	103
Continuing the Left Work .....	103
Last Word .....	104
<b>Reference Cited.....</b>	<b>105</b>

## List of Tables

Table 1.1	List of research questions posed and solved in this dissertation .....	6
Table 2.1	Examples of the collected data .....	13
Table 2.2	Examples of different forms of ratings .....	13
Table 2.3	User-item rating dataset.....	15
Table 2.4	Summary of the state-of-the-art models used on model-based CF.....	23
Table 2.5	Summary of the popular techniques used for CF .....	24
Table 2.6	The advantages and the drawbacks of the main CF techniques .....	25
Table 2.7	A brief description of the evaluation properties .....	26
Table 2.8	Ground truth of a recommendation of an item to a user.....	28
Table 3.1	Classification of machine learning models.....	33
Table 3.2	Different categories of ML models .....	33
Table 4.1	The description of the preprocessing steps.....	42
Table 4.2	The description of the used datasets .....	45
Table 4.3	The averages evaluation criteria of NBCF and CANNBCF models on four datasets .....	49
Table 5.1	Description of the used datasets .....	67
Table 5.2	Average performance comparison of RACF and other similarity functions on MAE and RMSE for four datasets .....	75
Table 6.1	Summary of different Learning-based RSs and their implementation approaches, platform, evaluation, dataset and results .....	83
Table 6.2	Tools used to develop PAARS .....	90

## List of Figures

Figure 4.1	The framework architecture of CANNBCF .....	41
Figure 4.2	The architecture of the ANN used to predict ratings .....	44
Figure 4.3	Performance comparison of NBCF and CANNBCF on ROC for four datasets.....	48
Figure 4.4	Performance comparison of NBCF and CANNBCF on Precision vs Recall for four datasets .....	50
Figure 4.5	Performance comparison of NBCF and CANNBCF on ROC and Precision vs Recall for Book-Crossing .....	51
Figure 4.6	Performance comparison of NBCF and CANNBCF on ROC and Precision vs Recall for Jester .....	51
Figure 4.7	Performance comparison of NBCF and CANNBCF on ROC and Precision vs Recall for Last.fm .....	52
Figure 4.8	Performance comparison of NBCF and CANNBCF on ROC and Precision vs Recall for MovieLens.....	52
Figure 4.9	Performance of NBCF and CANNBCF on Accuracy for the four datasets.....	53
Figure 4.10	Performance of NBCF and CANNBCF on RMSE@ $K$ for the four datasets.....	54
Figure 4.11	Rating distribution of the datasets.....	56
Figure 5.1	Algorithm for Alignment Cost Computation.....	66
Figure 5.2	RACF algorithm for rating prediction .....	66
Figure 5.3	Performance comparison of RACF and other similarity functions on MAE@ $K$ for MovieLens dataset .....	70
Figure 5.4	Performance comparison of RACF and other similarity functions on RMSE@ $K$ for MovieLens dataset .....	70
Figure 5.5	Performance comparison of RACF and other similarity functions on MAE@ $K$ for Jester dataset.....	71
Figure 5.6	Performance comparison of RACF and other similarity functions on RMSE@ $K$ for Jester dataset.....	71
Figure 5.7	Performance comparison of RACF and other similarity functions on MAE@ $K$ for Last.Fm dataset.....	72
Figure 5.8	Performance comparison of RACF and other similarity functions on RMSE@ $K$ for Last.Fm dataset.....	72
Figure 5.9	Performance comparison of RACF and other similarity functions on MAE@ $K$ for Book-Crossing dataset.....	73

Figure 5.10	Performance comparison of RACF and other similarity functions on RMSE@K for Book-Crossing dataset.....	74
Figure 5.11	Performance comparison of RACF and other similarity functions on MAE@2 for the different datasets.....	76
Figure 5.12	Performance comparison of RACF and other similarity functions on RMSE@2 for the different datasets.....	76
Figure 6.1	Framework architecture of the proposed model .....	86
Figure 6.2	Framework architecture of PAARS web-based application.....	90
Figure 6.3	The user interface for an entry of a student preferences.....	92
Figure 6.4	The user interface for a list of recommended courses based on a personalized preferences of a student .....	93
Figure 7.1	The prototype of the model that fuses information from Social Network Sites .....	99

## Chapter 1: Introduction

### Introduction

Collaborative filtering (CF) is fundamentally characterized by recommender systems (RSs), which have recently attracted researchers' attention [1]. The advancements of information retrieval (IR) and information filtering (IF) gave birth to the early evolution of RSs during the last few years [2]. The main functionality of RSs can be seen as the social process of suggestions and recommendations which ultimately reduce the variety of selections and present more useful selections to users. It was thus rational considering recommender system as a subfield of IR. Yet, the mid-1990s did witness the birth of RSs as an independent field of study [1, 3]. The ever-increasing data about users and items and the emergence of machine learning approaches have motivated recent development of RSs. Since the development of RSs technology has been influentially driven by the exceeding use of the web [3], the richness of RSs applications forms a fertile ground for researchers to invest and build more robust applications [4].

CF can be defined as systems and software tools that automatically and effectively generate recommendations of the most suitable items to a target user by predicting a user's predilections and preferences [5]. The prediction process depends on the previous knowledge of user's interests, a description of items, and the interactions between users and items [6]. The objective of developing CF systems is to effectively reduce the overload of available selections by exposing users to the most relevant and suitable items and services from a variety of alternatives. These systems generate personalized and un-personalized recommendations by developing models that analyze users' data, opinions and behavior, and also analyze items' descriptions to strategically predict a user's preferences. Yet, they do not only analyze a user's data and opinions, but also analyze the opinions of users with similar preferences for augmenting prediction accuracy. The perceptive evolution of the CF arises from a phenomenon in which people seek recommendations from others who share the same interests or have the same preferences [1, 7, 8].

Broadly, CF approaches are: (1) neighborhood-based, (2) model-based, (3) graph-based and (4) hybrid-based. The availability of these different approaches gives CF developers and designers a chance to select the best technique for achieving their purposes. Each approach has some advantages and drawbacks. For instance, neighborhood-based approach suffers from the limited coverage problem, the sparsity problem, the cold-start problem, and the relative slowness in the prediction process, while model-based approach suffers from the essential need to tune the parameters, the relative cost of training phases, and the relative slowness in the training phase [9]. Several advanced machine learning

(ML) algorithms including Bayesian clustering [1, 10, 11], Latent Semantic Analysis [1, 12, 13, 14], Latent Dirichlet Allocation [15], Maximum Entropy [16], Boltzmann Machines [17], Support Vectors Machines [18], Singular Value Decomposition [19], Principal Component Analysis [19], rule-based models [3], Neural networks[5, 20], and Probabilistic recommendation approaches [7, 21] are introduced to address these problems.

With the rapid growth of ML algorithms, several CF applications have been developed in different domains. As a field of study, the dramatic evolution of ML solves problems that appeared when CF systems were first developed. Researchers claim that the main research focus of current CF research, especially in the big data era, is how to effectively use ML algorithms to overcome the limitations that CF systems unexpectedly experience [22]. Several researchers have developed CF systems based on ML algorithms in large commercial systems such as Netflix [23], Amazon [12, 24], TripAdvisor [25], Jester [26], GroupLens [27], MovieLens [28], last.fm [29], Google News [30], Pandora [31], and YouTube [32], while CF techniques also can be used in complex, advanced, and real-time systems such as CF system for medical diagnosis [33], and a stock market portfolio CF system [34]. Moreover, interesting applications of CF based on ML algorithms support collaborative e-learning in the domain of e-education [35]. Broadly, CF systems based on ML algorithms can be used in several domains to recommend items and services to users, for instance e-commerce, e-learning, e-library, e-business, e-tourism, e-government and e-resource services [36, 37].

The structure of this chapter is organized as follows. It first discusses the problem statement and research motivation. It then discusses the scientific contribution and significance. Then, it discusses the research objectives and research questions. It finally discusses the organization of this dissertation.

### **Problem Statement and Research Motivation**

This experimental research comprehensively investigates the CF and examines the performance of the proposed models to address the data sparsity and limited coverage problems of the conventional CF systems. Using the proposed models to develop a robust CF system is extremely valuable for both academia and industry and is becoming increasingly crucial for recommending items and services to users.

ML algorithms have been introduced to address the following problems in CF: the data sparsity of rating matrix and the limited coverage [9]. While the data sparsity means that the user-item rating matrix is sparse because not all ratings have been provided by all users, the limited coverage problem means that the prediction and the similarity computations depend on the collaboratively rated items. Therefore, the aim of this empirical research is to develop and introduce models that address these problems. The

proposed models are examined and evaluated over a spectrum of application domains that recommend: (1) books, (2) movies, (3) music, (4) jokes. The selection of these domains is motivated by [39], since the researchers claim that the most used datasets are from these domains. These datasets are “Jester,” “MovieLens,” “Book-Crossing”, and “Last.fm”. The availability of these datasets on the Web and the typical use by developers and researchers in CF techniques motivate the selection of these datasets.

The following sections discuss the current conducted research in the field of CF to determine the main motivations of the present research. Xiangnan et al. introduced a general framework NCF short for Neural network-based CF [6]. Their framework expresses and generalizes a matrix factorization by utilizing a multi-layer perceptron to learn the user–item interaction function. The conclusion of their paper was that using deeper layers of neural networks is empirically approved to offer better recommendation performance.

Koren and Bell presented a paper that addresses the recent development of model-based CF systems [40]. They started their paper by introducing the different types of collected data and knowledge. Then, they discussed the latent factor models (a.k.a., model-based or learning-based algorithms). Also, they claimed that the Netflix Prize competition that happened in October 2009 has provided much of the recent progress in the field of CF. Also, they pointed out that it was the first time the research community gained access to a huge industrial dataset which consequently attracted thousands of researchers, scientists, enthusiasts, and engineers to the field of CF. They also discussed two categories of model-based techniques: (1) matrix factorization and (2) neighborhood-learning models with global optimization techniques that allow lifting the limit on neighborhood size. Suvash et al. proposed AutoRec, a novel autoencoder framework for CF [41]. They presented an efficiently trained model which outperforms the state-of-the-art CF techniques including: (1) biased matrix factorization, (2) Restricted Boltzmann Machines and (3) Local Low-Rank Matrix Approximation.

A study conducted by Hanwang et al. [42] addressed a major problem in CF that concerns calculating the similarity for items and users. They claimed that the current hashing methods for CF that exploit binary code learning procedures are still encountering the challenges of discrete constraints. Therefore, they proposed a principled CF hashing framework that hashes users and items as latent vectors in the form of binary codes, so that user-item affinity can be efficiently calculated in a Hamming space.

Other research [9, 40] discussed the most common rating normalization schemes such as mean-centric and Z-score. According to [9], when users assign a rating to an item, each user may differently perceive the rating process. Hence, users might be unwilling to give high or low rating value to items they prefer or do not prefer. Rating normalization process transforms users’ ratings to a general unified scale.



Researchers in [43, 44] examined the optimal number of neighbors when calculating predictions. They claimed that the optimal number of neighbors ranges from 20 to 50. However, other researchers argued that the optimal value of  $k$  (number of neighbors) should be estimated by cross-validation [9].

Researchers in [45] deliberated a way to transform the rating data to be suitable for supervised ML algorithms. They discussed a transformation technique that transforms user-item data from CF problem to a typical supervised learning. They presented a domain-independent transformation from user-items ratings matrix representation to a supervised learning dataset that supports supervised learning techniques to be effectively used for CF. In addition, they pointed out that their experiments have proven that the transformation, combined with supervised learnings approaches, has significantly outperformed classical CF methods.

Researchers have addressed several major aspects related to CF including its functions and needs [1], data and knowledge sources [1], fundamental techniques and methods [2, 9, 44, 47, 48, 49, 77], evaluation and assessment [50, 51, 52], applications and case studies [36], human decision making and user interfaces [53], privacy [54], trust [55], personality [56], and robust collaborative filtering [57]. However, none of these research examine the performance of using a hybrid model that uses clustering and artificial neural networks (ANNs) to address the data sparsity problem in the conventional CF.

Moreover, the current study indicates that there is a current research gap in addressing the data sparsity and limited coverage issues in CF. Hence, the present study addresses this problem by providing novel models. These models are based on ML algorithms. The main advantages of these models are their efficiency in addressing the identified issues and applicability to be used in different CF domains (e.g., music, movie, or book).

CF is still a fertile ground that allows researchers and scientists to conduct more studies and experiments to meet the needs of users, to handle the problems with big data and information overload, and to enhance customer relationship-management [36]. In view of this, this research points out that several CF research and papers have been conducted and published in the last decade. These papers address several aspects of CF systems. However, there is no existing research that addresses the problem of limited-coverage in CF using auxiliary information from SNSs. Therefore, this research introduces a novel model to integrate information from SNSs to augment the quality of CF using a hybrid model of content-based and collaborative-based RSs.

### **Scientific Contribution, Significance, and Impact**

This empirical research introduces three novel models to address the current limitations in the conventional CF. The first novel model uses ANNs and clustering algorithms to address the sparsity

problem. The applicability of this novel model is significant for providing better quality user experiences. The current research experimentally examines the use of ANNs and clustering and shows its superiority to solve the sparsity issue that the conventional CF technique experiences. The results of the experiments show that the proposed model effectively solves the sparsity issue, improves the quality of recommendations, and demonstrates promising prediction accuracy.

Addressing the limited-coverage problem is another significant scientific contribution to the field of CF. This research introduces a novel similarity model to address this issue. The proposed similarity model identifies regions of similarity between a pair of users in the rating matrix to compute the similarity between users. Based on these identified similar regions, CF computes the similarity scores between users and items to predict an active user's ratings on unseen items. There are two main advantages of the proposed similarity model. First, the prediction process is not sensitive to the number of k-nearest neighbors. The second advantage is that the similarity computation does not only consider co-rated items to compute the similarity scores in relatively sparse datasets, but it also considers aligning the un-co-rated items of users to classify relevant neighborhoods and generate recommendation. Based on the evaluation results, this research claims that the proposed similarity model can demonstrate promising prediction accuracy and improve the prediction performance of CF.

Moreover, this empirical research introduces a model that incorporates social network influencers' data to augment CF. It collects auxiliary information from SNSs. It is developed based on three main techniques. The first one is the content-based technique used to recommend items to an active user based on his/her previously collected data or interests. The second technique is a Bayesian classifier used to learn an active user's profile and to determine social network influencers who can contribute to the prediction process. The third technique is used to identify a set of social network influencers.

Besides the previous proposed models, this research provides a web-based framework used to provide course recommendation to student at University of Idaho. This framework automates the conventional advising process when students seek help from their advisors in the course-selection task. It helps the students to personalize their learning objectives and guide them to make smart selections based on their personal preferences. This framework helps both students and their advisors to discover more materials and courses that they may not know about it. It also facilitates the course-registration process for students. For instance, some students may have an academic hold on their account because they did not contact their advisor to decide which courses can be taken. Therefore, this framework helps students to find a list of courses related to their learning objectives to personalize the education process. Also, advisors and professors also can benefit from this framework by providing syllabuses and course details to students instead of sending emails. This framework also automatically connects each advisor to

his/her students since each provided advisory recommender system allows both advisors and students to create accounts. While students have specific features such as finding courses, adding these courses to a registration cart, and then sending a registration request with their list of courses to their assigned advisor, advisors have their own features such as adding new course, deleting course, and accepting or rejecting students' registration requests.

The summary of these scientific and technological contributions to the field of CF includes: (1) models that yield better quality recommendation, (2) models that address the data sparsity and limited coverage issues, (3) models that result in more accurate predictions, and (3) academic advisory recommender framework based on a web application.

### Research Objectives and Questions

The particular objectives of this experimental dissertation are: (1) providing a novel model based on clustering and ANNs that addresses the sparsity problem in CF, (2) introducing a novel similarity model that yields better accuracy results and enhances the recommendation quality when predicting rating for new items, (3) introducing a model to incorporate information from SNSs to augment CF, and (4) developing a framework based on a web application that automates the process of academic advisory task. The research questions posed in this dissertation are presented in Table 1.1.

Table 1.1 List of research questions posed and solved in this dissertation

No.	Research Question	Chapter
Q1	How could the use of clustering and ANNs solve the data sparsity issue in CF?	4
Q2	Does the use of different datasets collected from popular domains present a significant difference in the accuracy of recommendation prediction when applying clustering and ANNs in CF?	4
Q3	How could the use of rating alignment-based similarity model solve the limited coverage issue in CF?	5
Q4	Does the use of different datasets collected from popular domains present a significant difference in the accuracy of recommendation prediction when using rating alignment-based similarity model in CF?	5
Q5	Can the academic advisory task related to courses recommendation be automated using CF?	6
Q6	How to fuse and integrate social network influencers' data to augment the performance of CF?	7

## Thesis Statement

Developing models based on clustering and ANNs, rating alignment-based similarity function, and social influencers' data to address data sparsity and limited coverage issues in CF significantly improve the performance in terms of rating prediction and ranking accuracy as well as recall and F1-score.

## List of Publications

### *Author's Related Publications*

- **A. Althbiti** and X. Ma, "Social Network Influencers' Data Augmenting Recommender Systems,". *Proceedings of the 2020 International Conference on Computational Science and Computational Intelligence, Las Vegas, VA. In Press.*
- **A. Althbiti** , R. Alshamrani, and X. Ma, "A Literature Review of Data Mining Techniques Used in Collaborative Filtering Recommender Systems,". *Proceedings of the 2020 International Conference on Computational Science and Computational Intelligence, Las Vegas, VA. In Press.*
- **A. Althbiti** , R. Alshamrani, T. Alghamdi, S. Lee, and X. Ma, "Addressing Data Sparsity in Collaborative Filtering Based Recommender Systems Using Clustering and Artificial Neural Network,". *Proceedings of the 2021 Computing and Communication workshop and Conference, In Press.*
- **A. Althbiti** , S. Algarni, T. Alghamdi, and X. Ma, "A Personalized Academic Advisory Recommender System: A Case Study,". *Proceedings of 2021 The Asia Pacific Computer Systems Conference, In Springer.*
- **A. Althbiti** and X. Ma, "Collaborative Filtering," in *Encyclopedia of Big Data*, L. A. Schintler and C. L. McNeely, Eds. Cham: Springer International Publishing, 2019, pp. 1–4.
- **A. Althbiti** and X. Ma, "Machine Learning," in *Encyclopedia of Big Data*, L. A. Schintler and C. L. McNeely, Eds. Cham: Springer International Publishing, 2020, pp. 1–5.
- **A. Althbiti** , R. Alshamrani, T. Alghamdi, H. Ghanem, and X. Ma, "An Efficient Similarity Measure Based on Rating Alignment to Augment Collaborative Filtering ," *IEEE Access*, Under Review.

### *Author's Collaborative Papers not directly related to this Dissertation*

- R. Alshamrani, **A. Althbiti**, Y. Alshamrani, F. Alkomah, and X. Ma, "Model-Driven Decision Making in Multiple Sclerosis Research: Existing Works and Latest Trends,". In *Patterns*, 1(8), 2020, 100121.

### **Presentations in Conferences**

- Present two research papers at the IEEE International Conference on Computational Science and Computational Intelligence 2020 (CSCI 2020: December 16-18, 2020, Las Vegas, USA)
- Present a research paper at the IEEE 11<sup>TH</sup> Annual Computing and Communication Workshop and Conference (CCWC 2021: January 27-30, 2021)
- Present a research paper at the 2021 Asia Pacific Computer Systems Conference (APCS 2021: March 11-14, 2021)
- Present a poster at US2TS Semantic Technologies Symposium 2019, Duke, NC.
- Present two posters at University of Idaho Research Computing and Data Science Symposium, May 15, 2019.

### **Grants**

- US2TS 2018 travel grant awarded to attend US2TS Semantic Technologies Symposium 2018, Duke, NC.
- US2TS 2019 travel grant awarded to attend US2TS Semantic Technologies Symposium 2019, Duke, NC.
- A grant provided from the Department of Computer Science at University of Idaho to attend CSCI 2020 conference.

### **Organization**

The remainder of this dissertation is organized as follows:

- (1) Chapter 2 presents a literature review on CF-based recommender systems approaches and evaluation. It demonstrates that as a result of the ever-increasing number of available items in e- services, users have been overwhelmed. Therefore, it is essential to develop and apply algorithms to address the challenge of selection overload. CF systems have been developed to help users to find what they might be interested in among a range of available selections. CF systems have been widely discussed as an efficient approach to cope with the selection overload issue. This chapter presents a review of the CF techniques and data mining algorithms used for CF. Moreover, it discusses the evaluation metrics used to assess the performance of recommendation algorithms.
- (2) Chapter 3 discusses advanced ML approaches that intrinsically form the model-based CF systems. ML addresses the current research needs of how to build models that learn automatically through experience. It is a fast-evolving and multidisciplinary field between computer science and statistics, and it forms an essential bridge between artificial intelligence

- and data science. This chapter delineates how the emergence of learning approaches and the explosion of big data motivate the recent development of ML models. It surveys the fundamental ML algorithms and discusses how ML algorithms can alleviate the issue of dimensionality and offer solutions to automate prediction and detection.
- (3) Chapter 4 discusses the use of clustering and ANNs to address the data sparsity problem. This chapter proposes a novel model that uses clustering and artificial neural network to address the issue of data sparsity in CF. The proposed model Clustering and Artificial Neural Network Based Collaborative Filtering (CANNBCF) is evaluated using four different datasets from four popular domains (books, music, jokes, and movies). The proposed model is more effective than other models at solving the sparsity issue that the traditional CF technique encounters. In this chapter, intensive experiments are conducted to evaluate the performance of CANNBCF. The evaluation criteria include accuracy, precision, recall, F1-score, and Receiver Operating Characteristics used to examine the proposed model. The results of the experiments show that CANNBCF effectively solves the sparsity issue, improves the quality of recommendations, and demonstrates promising prediction accuracy.
  - (4) Chapter 5 discusses the use of rating alignment-based similarity model in CF to address the problems of data sparsity and limited coverage. To improve the accuracy and the quality of a CF system, this empirical research proposes a novel similarity model that identifies regions of similarity between a pair of users in the rating matrix to compute the similarity between users. Based on these identified similar regions, CF can compute the similarity scores between users and items to predict an active user's ratings on unseen items. There are two main advantages of the proposed similarity model. First, the prediction process is not sensitive to the number of  $k$ -nearest neighbors. The second is that the similarity computation does not only consider collaboratively rated (co-rated) items to compute the similarity scores in relatively sparse datasets, but it also considers aligning the un-co-rated items of users to classify relevant neighborhoods and generate recommendation. Moreover, the proposed similarity model can address the main drawbacks of the conventional similarity functions used in CF systems, including data sparsity and limited coverage problems. To validate the efficiency of the proposed similarity model, intensive experiments are conducted to compare the performance of the proposed model with other state-of-the-art similarity models using four datasets collected from four popular RSs' domains (music, jokes, books, and movies). Results retained from MAE and RMSE reveal that the proposed similarity model can demonstrate promising prediction accuracy and improve the prediction performance of CF. Also, the proposed model can effectively address the data sparsity and the limited coverage problems.

- (5) Chapter 6 discusses a personalized academic advisory recommender system developed for University of Idaho students. It argues that the development of CF introduces content-based filtering which does not need users' rating for items used to compute the similarity between users or items in CF. Also, it states that the traditional system of recommending courses to students is time-consuming, risky, and monotonous work, which negatively affects both student performance and the learning experience. While content-based filtering can introduce a solution to automate the process of course selection, this chapter introduces a personalized academic advisory RS (PAARS) that recommends a list of courses based on each student's profile and similar students' profiles. The primary data mining technique used to learn profiles for students is a  $k$ -nearest neighbors ( $k$ NN) classifier. The objectives of this chapter are twofold. The first is to introduce a model that personalizes the learning process, since each student may have different objectives than other students. Secondly, it aims to introduce PAARS web-based framework that automates the process of course recommendation. PAARS would help students to enhance their academic performance and improve their level of loyalty to their universities as well.
- (6) Chapter 7 discusses a novel model that fuses auxiliary information from SNSs to augment the performance of CF. It argues that the ever-increasing use of social network sites and the availability of Internet services have introduced opportunities for users to communicate and connect with one another. Social network sites can identify the similarity between users to recommend new friends to a user. Therefore, users might not only communicate with their actual offline friends, but also be motivated to communicate with strangers and friends of their actual friends. This chapter introduces a model that incorporates social network influencers' data for augmenting recommender systems. This model is developed based on three main techniques. The first one is the content-based technique used to recommend items to an active user based on his/her previously collected data or interests. The second technique is a Bayesian classifier used to learn an active user's profile and to determine social network influencers who can help augment the quality of recommendations. The third technique is used to identify social media influencers. This model can be generalized to other domains that collect a side information from external sources.
- (7) Chapter 8 presents the conclusions, limitations, and future research directions. It puts all the proposed models in their proper perspective.

## Chapter 2: A Literature Review on Model-Based Collaborative Filtering

*This chapter is based on:*

**A. Althbiti**, R. Alshamrani, and X. Ma, “A Literature Review of Data Mining Techniques Used in Collaborative Filtering Recommender Systems,”. *Proceedings of the 2020 International Conference on Computational Science and Computational Intelligence, Las Vegas, VA. In Press.*

**A. Althbiti** and X. Ma, “Collaborative Filtering,” in *Encyclopedia of Big Data*, L. A. Schintler and C. L. McNeely, Eds. Cham: Springer International Publishing, 2019, pp. 1–4.

### Introduction

Information retrieval (IR) is a major concept in the field of computer science and information technology. IR is an automated process that provides users with related information that meets their needs [92]. Users seek to find useful information among a large and diverse quantity of stored data. IR systems are utilized to provide solutions to overcome the problem of overwhelming data. The concept of IR encompasses several areas. These areas include, but are not limited to, collaborative filtering (CF) [59], information filtering [2], data mining [61], data retrieval [62, 63], content filtering [5, 20, 68], and recommender systems (RSs) [3, 5].

CF systems are software tools that automatically and effectively generate recommendations to users [1, 5, 59]. These useful recommendations assist users by effectively reducing large and diverse results that systems retrieve. The fundamental prediction models of CF are classified into three main categories:

- (1) Models to predict social connections between users in social platforms such as Twitter [72] or Facebook [3].
- (2) Models to predict ratings of a user for an item such as music [73], books [74], movies [47], or news [75]. This work only introduces the CF for the purpose of recommending items to users.
- (3) Models to rank the top- $K$  items for an active user [9] when only the ratings of a small list of items are available.

The main components of CF are the stored data, algorithms, and the filtered results. The stored data are related to users (e.g., user’s demographic factors or ratings for previous selected items) and items (e.g.,



item's descriptions or ratings provided by users). The term "item" denotes the retrieved results. The term "user" denotes to whom the recommendations are being generated.

The structure of this chapter is as follows. It discusses the potential needs of CF and its functions, the different types of collected data, several applications of CF, the main techniques of CF, the evaluation techniques used to assess the performance of CF, and finally the summary, respectively.

### **The Potential Needs of CF**

There are several things CF must accomplish, including:

- Reduce the number of retrieved results.
- Introduce a reliable system.
- Gain user's trust.
- Increase use's satisfaction and fidelity.
- Enhance the effectiveness of the retrieved results.
- Increase the number of purchased items [85] and sell more diverse items, especially in the e-commerce systems.

CF can perform several tasks that Herlocker et al. list in their significant work [43]:

- Understand user's needs and wants.
- Provide good recommendations.
- Annotate the recommendations to better present them.
- Explain why these recommendations are suitable for each user.
- Recommend a bundle which provides users with set of items that fit together (e.g., travel recommendation system [79]).

### **Types of Collected Data**

When developing a CF system, it is essential to consider what types of data sources and knowledge to collect, what algorithms to use, and what evaluation techniques can measure the performance and the accuracy. This section discusses the different types of collected data from users and items.

Broadly, the primary data are ratings, tags or reviews that come from users when evaluating items. More sophisticated CF requires collecting more data and knowledge about users and items. This acquisition is done through building semantic CF that uses ontological descriptions of users and items [81]. Other versions of CF may ask users to provide personal information to build a user profile that enhances the quality of the recommendation [9]. Previous studies have introduced a CF that considers

social network links or social relationships between users to augment CF's functionality and quality [83].

The three primary kinds of collected data are items, users, transactions. Item is a broad term that includes services, products, movies, music, new, etc. The data about an item are not limited to the item's description, attributes, or facts [24]. Furthermore, each item has its own value or utility. The item utility means the polarity of user's consumption. It can be positive when a user is willing to use it or negative when a user avoids using an item. Table 2.1 presents examples of the potential data collected about the items. CF designers and developers consider that users are not identical. Each user has his/her preferences, characteristics, goals, and objectives when interacting with a CF system [84]. Based on the used techniques, the collected data and knowledge about users may be organized in various ways [87]. It is noteworthy that the ability of CF to personalize recommendations heavily depends on exploiting a range of knowledge acquisition processes. Table 2.1 presents examples of the potential collected data about users.

Table 2.1 Examples of the collected data

	<b>Data about user</b>	<b>Data about item</b>
<b>Collected data</b>	Ratings, reviews, or user profile [92] (e.g., age)	Ratings, reviews, description, or tags

Transactions record what is happening between users and a CF system [1]. They comes in forms of ratings or reviews. These transactions build a history about users. They explicitly or implicitly track users' previous experiences, preferences, and behavior. The explicit collection is ratings provided directly by users when interact with a CF system. The implicit collection is indirectly collected or inferred by a CF system as a result of users' interaction. Table 2.2 presents an overview of ratings' forms [3].

Table 2.2 Examples of different forms of ratings

<b>Type</b>	<b>Example</b>	<b>Type of the collection</b>
<b>Numerical ratings</b>	1-5 stars	Explicit
<b>Ordinal ratings</b>	Strongly agree, agree, neutral, disagree, strongly disagree	Explicit
<b>Binary ratings</b>	Like vs dislike	Explicit
<b>Unary ratings</b>	The value 1 means a user purchased this items, the value 0 means no information recorded	Implicit

## CF Applications

The evolution of CF requires generous contributions from researchers in the fields of computer science, statistics, data science, machine learning, human computer interaction, deep learning, and data mining [22]. These contributions make CF a primary component of systems that collect data and eventually generate recommendations. For instance, CF has been used in large systems such as Netflix [23], Amazon [12, 24], TripAdvisor [25], Jester [26], GroupLens [27], MovieLens [28], last.fm [29], Google News [30], Pandora [31], and YouTube [32]. Moreover, CF can be used in complex, advanced, and real-time systems such as CF for medical diagnosis [33] and a stock market portfolio CF [34]. Other interesting applications support collaborative e-learning in the domain of e-education [35].

## CF Techniques

There are four main techniques classified by CF: (1) neighborhood-based CF (NBCF), (2) model-based CF (MBCF), (3) graph-based CF (GBCF) and (4) hybrid-based CF (HBCF). Researchers have introduced other terms for neighborhood-based technique such as memory-based CF [47].

### *Neighborhood-based CF*

As mentioned previously, users' ratings for previous experienced items are used to predict a rating for unexperienced items. This technique uses stored user-item ratings directly to make predictions. The process of rating predictions is slow. The reason is that each time the NBCF estimates predictions, it needs to access stored ratings, performs calculations, and then makes predictions. Researchers classify this technique as either (1) user-based CF or (2) items-based CF. The components of NBCF are: (1) ratings' normalization, (2) correlation computation (i.e., affinity or similarity computation), and (3) the process of neighbors' selection [61].

Starting with the ratings' normalization process, researchers consider that when users assign a rating to an item, each user may perceive the rating process differently [9]. Users might be not willing to give high or low rating value to items they prefer or do not prefer. Rating normalization process aims to transform users' ratings to a general and unified scale. The most common rating normalization schemes are mean-centric and Z-score.

The selection of the neighbors is an essential process which depends on the affinity between users or items. The quality of a recommendation and the accuracy of a prediction depend on the selection of similar neighbors. There are several similarity computation techniques available for use. Cosine Vector (CV) and Pearson's Correlation Coefficients (PCC) are popular measures for computing the similarity between two objects [9].

The number of selected neighbors plays a crucial role in the prediction process. It is not optimal to include ratings of every user in the process of predicting rating for an item of a target user. There are rules and standards to follow when selecting the number of neighbors. Researchers point out that filtering the neighbors should be carefully performed in two main filtering steps: (1) global filtering, which keeps similar users or items, and (2) a pre-prediction step, which only keeps best candidates to contribute to the prediction process. It is noteworthy that [43, 44] claim that the optimal number of neighbors ranges from 20 to 50. However, other researchers argue that the optimal value of  $k$  (number of neighbors) should be estimated by cross-validation [9].

### User-based CF

User-based technique uses previous ratings of users who share the interests as a target user to predict ratings for unseen items of a target user. User-based technique initially finds similar rating patterns among users and a target user. Then, it measures the similarity and finds neighbors of a target user. Finally, it uses neighbors' ratings to predict ratings for unseen items to a target user.

The goal of using user-based technique, as previously mentioned, is to predict ratings of a target user  $u$  for unrated item  $i$ . Researchers apply different algorithms to estimate these predictions [14,102]. For instance, Table 2.3 includes a user-item rating matrix which stores four users' rating for four items. The task is to predict a rating of *Mark* for unrated *item3*. In this example, a regression algorithm is used to compute a prediction by calculating a weighted average of *Mark's* neighbors' ratings.

Table 2.3 User-item rating dataset

	<b>i1</b>	<b>i2</b>	<b>item3</b>	<b>i4</b>
<b>Mark</b>	3	3	?	5
<b>U1</b>	4	2	2	4
<b>U2</b>	1	1	4	2
<b>U3</b>	5	2	3	4

To estimate a prediction of *Mark* for an *item3*, the following notations are provided. A set of users is symbolized as  $U = \{U1, \dots, Uu\}$ ; a set of items is symbolized as  $I = \{I1, \dots, Ii\}$ ; a matrix of ratings is symbolized as  $R$ , where  $r_{u,i}$  means rating of a user  $U$  for an item  $I$ ; and a set of possible ratings is symbolized as  $S$ , where its values take range of numerical ratings  $\{1,2,3,4,5\}$ . Most systems consider the value 1 as “strongly dislike” and the value 5 as “strongly like”. It is worth noting that  $r_{u,i}$  should only take one rating value.

The first step is to compute a similarity between *Mark* and the other three users. In this example, a similarity between users estimated using PCC (2.1) is given.

$$sim(u, v) = \sum_{i \in I} (r_{u,i} - \overline{r_u})(r_{v,i} - \overline{r_v}) / \sqrt{\sum_{i \in I} (r_{u,i} - \overline{r_u})^2} * \sqrt{\sum_{i \in I} (r_{v,i} - \overline{r_v})^2} \quad (2.1)$$

where  $\overline{r_u}$  and  $\overline{r_v}$  are the average rating of the available ratings provided by users  $u$  and  $v$ , and  $i \in I$  represents these items that user  $u$  and  $v$  previously both rated.

By applying (1) to the given ratings in Table2.3, given that ( $\overline{r_{Mark}} = \frac{3+3+5}{3} = 3.6$  , and  $\overline{r_{U1}} = \frac{4+2+2+4}{4} = 3$ ), the similarity between Mark and *U1* is estimated as follows:

$$sim(Mark, U1) = \frac{(3-3.6)(4-3)+(3-3.6)(2-3)+(5-3.6)(4-3)}{\sqrt{(3-3.6)^2+(3-3.6)^2+(5-3.6)^2} \times \sqrt{(4-3)^2+(2-3)^2+(4-3)^2}} = 0.49$$

It is noteworthy that values of PCC are in the range of (+1 to -1), where +1 means high positive correlation and -1 means high negative correlation. The similarities between *Mark* and  $\{U2 \text{ and } U3\}$  are 0.15 and 0.19, respectively. Referring to the previous calculations, it is claimed that *U1* and *U3* rated several items in the past in the same fashion as *Mark* did. Thus, *U1*'s and *U3*'s ratings are utilized to predict a rating of *Mark* for *item3*.

The second step is to compute a prediction for *item3* using the ratings of *Mark*'s  $K$ -neighbors (*U1* and *U3*). Thus, (2.2) is introduced where  $\hat{r}$  means a predicted rating.

$$\hat{r}(u, i) = \overline{r_u} + \frac{\sum_{v \in K} sim(u, v) * (r_{v,i} - \overline{r_v})}{\sum_{v \in K} sim(u, v)} \quad (2.2)$$

Then, equation (2) is used to calculate the prediction.

$$\hat{r}(Mark, item3) = \overline{r_{Mark}} + \frac{sim(Mark, U1) * (r_{U1, item3} - \overline{r_{U1}}) + sim(Mark, U3) * (r_{U3, item3} - \overline{r_{U3}})}{sim(Mark, U1) + sim(Mark, U3)} = 4.54$$

Given the result of the prediction 4.45, it is most likely that *item3* will be a good selection to be recommended to *Mark*.

Another approach in [100] predicts ratings using a classification technique. It considers ratings as classes and estimates a rating for an item  $i$  of an active user  $u$  by taking a vote of  $u$ 's neighbors. For instance, if  $u$ 's neighbors vote as a value 2 for an item  $i$ , then the classification result for the prediction is 2. Equation (2.3) estimates a prediction of a rating for an item of a target user. After calculating similarities between a user  $u$  and  $u$ 's neighbors, a vote  $v_{ir}$  for a rating is estimated as a sum of similarity weights [64].

$$v_{ir} = \sum_{v \in N_i(u)} \delta(r_{vi} = r) w_{uv} \quad (2.3)$$

where  $\delta(r_{vi} = r)$  is the value 1 if  $r_{vi} = r$ , otherwise the value is 0. Then, the predicted rating is a value  $r$  for which  $r_{vi}$  is the greatest value, after computing (3) for every rating value, in this specific case  $S = \{1, 2, 3, 4, 5\}$ .

### Item-based CF

Item-based technique uses previous ratings provided by a user to predict ratings for unseen or unexperienced items of the same user. There are two categories of item-based technique: (1) personalized item-based technique, where recommendations are generated differently for each user, and (2) non-personalized technique, where recommendations are generated in a form of top-10 selections for all users.

Item-based technique initially computes the similarity between items and identifies the similar neighbors of a target item. Then, it uses neighbors' ratings of an item to predict a rating of a target user [47]. In this technique, the actual ratings of a user are used to predict rating for a target item. Users may prefer to interact with systems that use their ratings instead of using other users' ratings. Equation (2.4) computes the similarity between two items.

$$sim_{(i,j)} = \sum_{u \in U} (r_{u,i} - \bar{r}_i) \cdot (r_{u,j} - \bar{r}_j) / \sqrt{\sum_{u \in U} (r_{u,i} - \bar{r}_i)^2} * \sqrt{\sum_{u \in U} (r_{u,j} - \bar{r}_j)^2} \quad (2.4)$$

where  $\bar{r}_i$  and  $\bar{r}_j$  are average ratings of available ratings made by a user for both items  $i$  and  $j$ . Then, estimating a prediction for an item  $I$  of a user  $U$  by applying (2.5), where  $K$  means the number of neighbors of items for item  $I$ .

$$\hat{r}(u, i) = \bar{r}_i + \frac{\sum_{j \in K} sim(i,j) * (r_{u,j} - \bar{r}_j)}{\sum_{j \in K} sim(i,j)} \quad (2.5)$$

### *Model-based CF*

In these techniques, machine learning (ML) algorithms and data mining (DM) models are used in the context of predictive models. It uses user-item rating matrix to learn a predictive or a parametric model (i.e., latent factor model) that explains relationships between users, items, or both. To learn a model that predicts ratings, a model should have a set of parameters that capture salient characteristics of users and items. These parameters are learned from training data and used to predict new ratings [47, 10]. MBCF is classified into two main categories: (1) factorization technique and (2) adaptive neighborhood learning technique [9, 40]. These models are introduced to address the main drawbacks in the conventional CF.

There are three fundamental models introduced to address the sparsity and limited coverage problems in CF: Matrix Factorization (MF), computational intelligence, and mathematical calculation [113]. This chapter thoroughly discusses these models and introduces the readers to the most used models in the literature. Moreover, chapter 3 discusses ML algorithms applied to CF in details.

### Factorization Technique

Factorization techniques are introduced as ML algorithms that solve the problems of data sparsity and large dimensionality space. When Netflix in 2009 offered a prize for the best system that yields better accuracy, several participating teams used matrix factorization, which effectively increases the predictive accuracy [89]. These techniques project users and items into a reduced latent space. The dimensionality reduction can capture high-level patterns in the data, explain the relationships between users, items, or both, and capture their most hidden characteristics [14, 99].

Transforming the large, complex, and high-dimensional matrix of user-item rating space into a smaller and organized space can alleviate the problem of data sparsity. Researchers identify two categories in which factorization can be utilized to improve the accuracy of predictions [9]. These categories are: (1) factorization of a sparse user-user or item-item similarity matrix, and (2) factorization of the actual ratings matrix.

#### (1) Factorization of a sparse user-user or item-item similarity matrix

The similarity matrix, which contains the similarity values calculated via PC or other similarity computation techniques, is sparse because a user may rate a few items, and an item might be rated by a few users. Researchers in [9] proposed a solution to densify a sparse similarity matrix that estimates a low-rank approximation using factorization techniques.

The transformation of the  $m \times n$  ratings matrix  $R$  into a lower-dimensional space is done using principal component analysis (PCA) or singular value decomposition (SVD). Moreover, the result is a matrix  $R'$  of size  $m \times d$ , where  $d \ll n$ . Therefore, this reduced, fully specified, and organized matrix allows the similarity between users, items, or both to be computed. This happens when using the determined  $d$ -dimensional representation of each user or each item to compute the similarity with a target user or item [5]. Hence, estimating the similarity using the reduced representation is more powerful because new transformed vectors are fully dense. Furthermore, computing the similarity in this reduced space is a simple task by utilizing cosine or dot product on the reduced vectors. The process of computing the low-dimensional representation involves the following steps.

The first step is to augment the  $m \times n$  ( $m$  represents user, and  $n$  represents items) sparse ratings matrix  $R$  by filling in the missing values. A detailed explanation of techniques to fill in the missing values is introduced in [88]. One basic technique to fill in the missing entries is to estimate the mean of the corresponding row or column, then replace the missing values with the mean. Hence, the matrix  $R_f$  is fully specified to be utilized to compute the similarity between pairs of items or users — in this specific section, between items  $n$ . Then, a symmetric  $n \times n$  similarity matrix, semi-definite, is computed between pair of items. This symmetric matrix is produced by  $A = R_f^T R_f$ .

The second step is to define the dominant basis vectors of  $R_f$  for SVD. This requires performing the diagonalization of the similarity matrix  $A$ .

$$A = P\Delta P^T \quad (2.6)$$

where  $P$  is a  $n \times n$  matrix whose columns contain the orthogonal eigenvectors of  $A$  [3]. The  $\Delta$  is a diagonal matrix that has non-negative eigenvalues of  $A$ . Assume that  $P_d$  is a  $n \times d$  matrix that contains only the columns of  $P$  corresponding to the largest  $d$  eigenvectors. Then, the low-rank approximation of  $R_f$  is estimated by the matrix product  $R_f P_d$ . Because  $R_f$  is a  $m \times n$  matrix and  $P_d$  is a  $n \times d$  matrix, the dimensions of the low-rank approximation matrix  $R_f P_d$  are  $m \times d$ . Hence, each of the  $m$  users is now approximated by a low-rank representation of  $d$ -dimensional space. This representation is utilized to find the neighbors of each user. Then, the neighbors' rating is utilized to predict an item rating of a target user with (2.7).

$$\hat{r}(u, i) = \overline{ru} + \frac{\sum_{v \in K} \text{sim}(u, v) * (rv, i - \overline{rv})}{\sum_{v \in K} \text{sim}(u, v)} \quad (2.7)$$

To use the above approach for item-based technique, the transposition of  $R_f$  is utilized in a similar fashion for the entire low-rank approximation representation. Researchers in [82] also investigated utilizing other dimension reduction techniques such as PCA. Broadly, the results of SVD and PCA are somewhat similar [3]. It is noteworthy that PCA technique utilizes the co-variance matrix of  $R_f$  instead of the similarity matrix  $R_f^T R_f$ .

## (2) Factorization of the rating matrix

Factorization of the rating matrix can address the problems of cold-start, where there are no ratings for users or items, limited coverage, and data sparsity. The same process described above is applicable here, except instead of utilizing the similarity matrix, the actual rating matrix is used.

### Adaptive neighborhood learning methods



The traditional way to determine a similarity between users, items, or both is through similarity commutation methods. Learning the neighborhood automatically from the ratings is possible because of the recent development in the field of MBCF. Ning et al. developed an interesting method based on the item-item regression [80]. This representative neighborhood-learning method is known as sparse linear neighborhood models (SLIM). The interesting part behind this algorithm is how the sparsity in the regression coefficients is encouraged. Then, SLIM utilizes regularization methods to learn the neighborhood parameters by minimizing the squared prediction error. Standard regularization and sparsity are determined by penalizing the  $l_2$ -norm and  $l_1$ -norm of the parameters. Combining these two regularization techniques in a regression problem is introduced in [71] as elastic net regularization. The following optimization problem is used for the learning process

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \frac{1}{2} \| R - RW \|_F^2 + \frac{\beta}{2} \| W \|_1 + \lambda \| W \|_2 \\ \text{subject to } & W \geq 0 \\ & \text{diag}(W)=0 \end{aligned} \tag{2.8}$$

where  $\beta$  and  $\lambda$  are the parameters that control the amount of each type of regularization. More advanced techniques based on SLIM are introduced in [69,70].

#### Solving the Data Sparsity Problem Using Advanced Models

Researchers in [110] proposed a novel model that addresses the data sparsity problem in CF. Their model makes use of the Linked Open Data (LOD). Then, LOD combines it with Matrix Factorization model. The researchers used DBpedia for the LOD to find enough information about new entities and applied Matrix Factorization to handle the data sparsity problem. They argued that the results are superior to other existing methods and their model presents a better recommendation accuracy. However, their model has one main limitation related to acquiring the additional information: they only use DBpedia to acquire additional information. Another limitation is the complexity of the model which ultimately increases the computation cost and space complexity. Future work can improve the proposed model by using LOD knowledge bases like Freebase, LinkedMDB, and YAGO and incorporating users' social relationships in online social networks to find enough auxiliary information. Also, Deep Learning techniques should be used along with Matrix Factorization to generate better recommendations.

Having a sparse user-item rating matrix is the main challenge in CF. Researchers proposed a model that integrates Social Balance Theory (abbreviated as SBT; e.g., "enemy's enemy is a friend" rule) in CF to put forward a novel data-sparsity tolerant CF approach. During the prediction process, a pruning

strategy is used to reduce the searching space for neighbors and to improve the recommendation efficiency [147]. They developed a mathematical framework that represents several rules and conditions of social relationships between users. Then, they computed the similarity between users to find a set of neighbors to a target user who receives recommendations. They conducted their experiments on a real web service quality dataset WS-DREAM which contains 1,974,675 records of response time over 5825 services from 339 users. Finally, their model was validated in terms of recommendation accuracy, recall and efficiency. Their results suggest that the use of pruning strategies and Social Balance Theory can improve the recommendation quality of CF.

Addressing the sparsity problem while considering privacy and security concerns is another research direction. Researchers have argued that several previous studies used hybrid approaches that incorporate auxiliary data sources into CF, like content, context, or social relationships [148]. They proposed a NBCF approach enhanced by a novel similarity reinforcement mechanism. Their proposed approach learns potential similarity relationships between users or items by making better use of identified but limited user-item interactions. Then, these identified interactions are used to make more reliable and accurate rating predictions. Their approach incorporates user similarity reinforcement and item similarity reinforcement into a combined framework and lets them augment each other. The researchers conducted intensive experiments on several public datasets and argue that their approach achieves a significant improvement in prediction accuracy when compared with the state-of-the-art NBCF and MBCF algorithms.

Developing similarity measures have been considered to address the data sparsity issue in CF. A novel similarity method to improve the accuracy of conventional CF under sparse data issue is discussed in [149]. The proposed similarity model is based on the global user preference and has the ability to solve the similarity issue by finding the relationship among non-correlated users. It identifies the optimal number of neighbors to a target user by using two main strategies: (1) fairness and (2) proportion of collaboratively rated (co-rated) items. The researchers argue that the accuracy of the developed method is improved compared to the conventional CF similarity functions using MAE, Recall, Precision, and F-score evaluation metrics.

Another study was carried out to address the data sparsity problem using three novel views of reliability measures [150]. The first is a user-based reliability measure used to assess the performance of users' rating profiles in predicting unseen items. While the second novel mechanism is used to augment the rating profiles with low quality by adding a number of reliable ratings, the item-based reliability measure is used as the second view of the reliability measures, and then a number of items with highest reliability values are added into a target user's profile. Therefore, these enhanced users' profiles are

used to estimate the similarity values between users and also initial ratings of unseen items. Finally, the process of recalculating unreliable predicted ratings is performed using rating-based reliability measure as the third view of the reliability measures. The researchers argue that the proposed method significantly outperforms other CF methods.

Table 2.4 demonstrates a list of models proposed to address the data sparsity in CF. It discusses the main characteristics of the models including the used method, the addressed problem, the evaluation, the advantages, the limitation and problems associated with each model, and finally the suggested future work and enhancements.

#### *Graph-based CF*

Graph-based techniques can alleviate the problem of data sparsity in the rating matrix. Using a structural transitivity while using graph models can define a similarity in the neighborhood-based techniques [3]. These graphs provide an organized representation to map the relationships between users, items, or both.

To represent the data, this technique forms a graph in which nodes are users or items, and links (edges) signify interactions, correlations or similarities between users and items [9]. There are mainly two ways to calculate the similarities between users or items: (1) path-based similarity [86] (i.e., shortest-path methods [3]), and (2) random walk similarity [64]. To calculate the similarity between users, items, or both, a user-item graph can be constructed in one of three ways: (1) user-item, (2) user-user, and (3) item-item graphs [3].

#### *Hybrid-based CF*

Hybrid techniques can be a combination of all the previous mentioned techniques of CF. Researchers introduce different techniques that combine two or more techniques to improve the prediction accuracy. Table 2.5 presents the main CF techniques and the popular ML and DM algorithms that can be used as a latent “semantic” factor models.

The availability of these different techniques gives CF designers and developers a chance to select the best technique that meets users’ needs. For instance, which is better to use: a user-based approach or an item-based approach? The answer is that if a system has 10 million users and 10 thousand items, it is better for the system to calculate similarities between items [43]. Hence, item-based technique is a good option.

Table 2.4 Summary of the state-of-the-art models used as model-based CF

Model	Method	Addressed Problem	Evaluation	Advantages	Limitations and Problems	Suggested Future Work
(RS-LOD and MF-LOD) [110]	Matrix Factorization	Solve data sparsity and reduce the dimensions	RMSE	Recommendation accuracy is improved and new similarity called LOD (Linked Open Data) is introduced	The method process is complex	Use Freebase, LinkedMDB, and YAGO to mine more constructive information
Data-Sparsity Tolerant Web Service Recommendation [147]	Apply Pruning strategies and Social Balance Theory	Solve data sparsity	Recall, MAE, and Time Cost	The search space while finding the possible friends is reduced	It does not consider context information	Use Social Poisson Factorization
Mitigating Data Sparsity Using Similarity Reinforcement-Enhanced Collaborative Filtering [148]	Incorporate user similarity reinforcement and item similarity reinforcement into a combined framework	Solve data sparsity and poor prediction problems	MAE and RMSE	Privacy and security aspects are considered	The interpretability of the evaluation measures is poor	Optimize the algorithm as well as explore the possibility to accelerate it by means of parallel computing
Similarity model [149]	Compute the similarity between users based on based on the global user preference	Solve data sparsity and find optimal number of neighbors	MAE, Accuracy, precision, Recall, and F1-score	The evaluation for both prediction accuracy and ranking accuracy is considered	It depends on co-rated items	Consider working on unclassified item datasets
A novel approach based on multi-view reliability measures to alleviate data sparsity in recommender systems [150]	Compute the similarity between users using problem using three novel views of reliability measures	Solve the data sparsity, improve the prediction	Normalized MAE and Normalized RMSE	Several models to generate reasonable recommendation are fused. The high prediction accuracy and high practicability to different domains that use CF are achieved.	The proposed framework of CF is too complicated and the computational complexity is high	Consider using the noise detection techniques to find most useful information in the user-item rating matrix

Table 2.5 Summary of the popular techniques used for CF

CF techniques		
<b>NBCF</b>	User-based CF [44, 48, 49, 60, 77]	Item-based CF [9, 44, 47, 60, 77]
<b>MBCF</b>	Bayesian clustering [10,11,59], Latent Semantic Analysis [12, 93], [11, 13, 14], Latent Dirichlet Allocation [15], Maximum Entropy [16], Boltzmann Machines [17], Support Vectors Machines [18], Singular Value Decomposition [18], Principal Component Analysis [19], rule-based models[3, pp. 71-138], Neural networks[5, 6], and Probabilistic recommendation approaches [7, 21].	
<b>GBCF</b>	[103]	
<b>HBCF</b>	[76, 96, 101]	

Likewise, when a system has fewer users than items, using user-based technique might be an optimal choice. Researchers in [9] addressed five main criteria when choosing between a user-based approach or an item-based approach. These criteria are (1) accuracy, (2) efficiency, (3) stability, (4) justifiability, and (5) serendipity [9]. Table 2.6 discusses the advantages and drawbacks of the previous introduced techniques.

### Evaluation Methods

The evaluation step is crucial to examine the accuracy of predictions. Also, the evaluation allows CF designers and developers to decide which algorithm outperforms other candidates when it comes to accuracy or other evaluation properties.

#### *Evaluation Settings*

There are three different settings of evaluation: (1) offline, (2) online, and (3) user studies [5]. It is noteworthy that these evaluation criteria are used in other related area such as ML and data retrieval.

- (1) In the offline setting, the main goal is to select the most powerful algorithm among other candidates. The evaluation is done using previous collected datasets. CF researchers and designers prefer this type of evaluation setting because it does not require direct interactions with users. Moreover, it allows several candidates of algorithms to be examined on the same dataset. Also, it allows several datasets to be examined using the same algorithm. This type of evaluation can assess a prediction power of an algorithm. However, it cannot evaluate the whole process of interactions [3].

Table 2.6 The advantages and the drawbacks of the main CF techniques

CF techniques	Advantages	Drawbacks
<b>NBCF</b>	Simplicity, justifiability [99], efficiency, stability, no need to have a training phase, and learning fast.	Limited coverage, sensitivity of the sparsity, cold-start problem, and prediction process is slow.
<b>MBCF</b>	Solving the limited coverage problem, solving the problem of sparsity in the rating matrix, can capture salient characteristics in the data, more robust to outliers, good option for a model generalizability [9], yielded greater accuracy and stability [15], requires less memory compared to other approaches, and prediction process is fast.	Parameters need to be tuned, costly training phases are required, and the training process is slow.
<b>GBCF</b>	Solving the limited coverage problem and solving the problem of sparsity in the rating matrix.	The space and the time complexity when calculating the similarities for large commercial applications, complexity, and the adoption of these systems is costly.
<b>HBCF</b>	Better recommendations, high accuracy	The space and the time complexity when calculating the similarities for large commercial applications, complexity, and the adoption of these systems is costly.

- (2) In the online setting, the main goal is to evaluate the behavior of users when interacting with different candidates of algorithms. These algorithms are deployed online and allow users to directly interact. Recording the transactions between users and items allows CF designers to evaluate which algorithm outperforms other candidates.
- (3) In the user studies setting, the main goal is to directly evaluate a set of algorithms by asking users to interact with different candidates of algorithms. While users use these systems, their behavior is observed and recorded [93]. Also, those recruited users may be invited to take a

survey and answer questionnaires. They may be asked qualitative and quantitative questions before, during, and after their interaction with the system [94, 95].

### *Evaluation Properties*

There is a range of evaluation criteria that are widely used when deciding which algorithm to use [3]. They are: (1) accuracy, (2) coverage, (3) confident, (4) trust, (5) novelty, (6) serendipity, (7) utility, (8) risk [66], (9) robustness, (10) privacy, (11) adaptivity [67], and finally (12) scalability [66]. Table 2.7 demonstrates these different properties.

Table 2.7 A brief description of the evaluation properties

<b>Evaluation property</b>	<b>A brief description</b>
<b>Coverage</b>	Evaluates the ability of an algorithm to recommend most of items to users [52].
<b>Confidence</b>	Evaluates the ability of an algorithm's trust in its recommendations or predictions [51].
<b>Trust</b>	Evaluates the ability of an algorithm to gain users' trust [50].
<b>Novelty</b>	Evaluates the ability of an algorithm to recommend items that have not been experienced by users [52].
<b>Serendipity</b>	Evaluates the ability of an algorithm to recommend interesting items that might not have otherwise discovered by user [52].
<b>Utility</b>	Evaluates the ability of an algorithm to estimate how useful an item is to a user [98].
<b>Risk</b>	Evaluates the ability of an algorithm to avoid or minimize the risks that might occur if a user utilizes a recommended item [97].
<b>Robustness</b>	Evaluates the stability of an algorithm in the presence of fake information or ratings that aim to influence and bias the recommendations [91].
<b>Privacy</b>	Evaluates the ability of an algorithm to protect the information and ratings provided by users from attacks and malicious transactions [90].
<b>Adaptivity</b>	Evaluates the ability of an algorithm to be operated in different sittings where item collections may rapidly change [67].
<b>Scalability</b>	Evaluates the ability of an algorithm to process recommendations of large collection of items for large population of users [66].

### Accuracy

Accuracy is the property most frequently used to gauge the performance of different CF algorithms [52]. There are two classes for measuring the accuracy: (1) an accuracy of the ratings predictions, when ratings are available where an algorithm's objective is to learn function  $f: \mathcal{U} \times \mathcal{I} \rightarrow \mathcal{S}$  that predicts a rating of a user  $u$  for an item  $i$ , and (2) an accuracy of the usage predictions [5], when the objective of

CF is to recommend items as the top- $K$  items that may have a utility for users. It is noteworthy that when researchers evaluate algorithms, they use a different set of instances for evaluation step known as the testing-set, while they use other instances to train algorithms known as the training set.

(1) Measuring an accuracy of ratings prediction

These types of accuracy metrics are utilized to evaluate a prediction accuracy when estimating a rating of a user for an item. Examples of these measurements are:

- Root mean squared error (RMSE) is a popular metric given by (2.9),

$$RMSE = \sqrt{\frac{1}{|\mathfrak{I}|} \sum_{(u,i) \in \mathfrak{I}} (\hat{r}_{ui} - r_{ui})^2} \quad (2.9)$$

where,  $|\mathfrak{I}|$  is the set of recommended items  $i$  from a test set,  $\hat{r}_{ui}$  is a predicted rating for item  $i$  of a target user  $u$ , and  $r_{ui}$  is an actual rating for an item  $i$  of a target user  $u$  from a training set.

- Mean absolute error (MAE) is another alternative metric given by (2.10),

$$MAE = \frac{1}{|\mathfrak{I}|} \sum_{(u,i) \in \mathfrak{I}} |\hat{r}_{ui} - r_{ui}| \quad (2.10)$$

where,  $|\mathfrak{I}|$  is the set of recommended items  $i$  from a test set,  $\hat{r}_{ui}$  is a predicted rating for item  $i$  of a target user  $u$ , and  $r_{ui}$  is an actual rating for an item  $i$  of a target user  $u$  from a training set.

- Mean squared error (MSE) is given by (2.11),

$$MSE = \frac{1}{|\mathfrak{I}|} \sum_{(u,i) \in \mathfrak{I}} (\hat{r}_{ui} - r_{ui})^2 \quad (2.11)$$

where,  $|\mathfrak{I}|$  is the set of recommended items  $i$  from a test set,  $\hat{r}_{ui}$  is a predicted rating for item  $i$  of a target user  $u$ , and  $r_{ui}$  is an actual rating for an item  $i$  of a target user  $u$  from a training set.

Measuring accuracy does not always give a good estimate of the models' performance [65]. For instance, consider a binary class dataset where 98% of the data falls into one class, and 2% of the data goes to the other class. Now, assume that an algorithm wants to predict a class for a data point: it would have 98% accuracy. Therefore, this accuracy does not mean this algorithm is performing well. Hence, measuring the accuracy of estimating ranking is introduced.

(2) Measuring an accuracy of usage prediction (a.k.a., measuring the accuracy of estimating ranking)



This type of evaluation is often done in the offline setting where historical data are used to evaluate proposed methods [18]. As mentioned previously, this process does not evaluate the accuracy of the predictions; rather, it evaluates how an algorithm provides estimates of the underlying ranks [3], or it evaluates the actual consumption of recommended items. The evaluation is based on the ground-truth. These evaluation methods are designed for systems that collect unary or implicit ratings such as Netflix.

Researchers in [3] classify an accuracy of usage prediction based on nature of the ground-truth into three categories: (1) rank-correlation measures (RCM), (2) utility-based measures (UBM), or (3) receiver operating characteristics (ROC). The latter two techniques are used for unary datasets (implicit ratings).

This evaluation is performed on datasets that consist of users and their preferred items. Then, a ranking accuracy of a CF is evaluated after hiding some of a test user's selections. Table 2.8 presents the four possibilities of ranking outcomes (a.k.a., confusion matrix, or ground-truth). Examples of these measurements are:

- Precision (a.k.a., positive predictive value)

The precision is a ratio of the number of true positive recommended items to all the recommended items, given by (2.12),

$$Precision = \frac{tp}{tp+fp} \quad (2.12)$$

- Recall (a.k.a., true positive rate, sensitivity)

The recall is a ratio of the number of true positive recommended items to all the consumed items, given by (2.13),

$$Recall = \frac{tp}{tp+fn} \quad (2.13)$$

Table 2.8 Ground truth of a recommendation of an item to a user

	<b>Recommended</b>	<b>Not recommended</b>
<b>Consumed</b>	True-positive (tp)	False-negative (fn)
<b>Not consumed</b>	False-positive (fp)	True-negative (tn)

- Specificity (a.k.a., true negative rate)

The specificity is a ratio of the number of true negative not recommended items to all not consumed items, given by (2.14),

$$\text{Specificity} = \frac{tn}{tn+fp} \quad (2.14)$$

- False Positive Rate (1-Specificity)

The false positive rate is a ratio of the number of false positive recommended items to all the not consumed items, given by (2.15),

$$\text{False Positive Rate} = \frac{fp}{fp+tn} \quad (2.15)$$

- Area under the receiver operating characteristics curve (AUC)

AUC measurement is a useful measurement for comparing several algorithms independently of application [3].

- F-score

F-score is the harmonic mean between the precision and the recall and reveals a better quantification than either the precision or the recall [3] and is given in (2.16),

$$F = \frac{2*precision*recall}{precision+recall} \quad (2.16)$$

- Discounted cumulative gain (DCG)

DCG measures the usefulness of an item based on its location in a recommendation list of a specified length [64]. Also, it measures the effectiveness of algorithms that rank items.

## Conclusion

This chapter extensively discusses important perspectives of CF. It discusses the fundamental tasks of CF and its functions. Then, it sheds light on a different type of required data to develop a CF system. It thoroughly covers CF techniques, including neighborhood-based CF, model-based CF, graph-based CF, and hybrid-based CF. Model-based technique can address the limitations of neighborhood-based CF. Moreover, this chapter considers a number of the state of the art CF techniques. Finally, evaluation techniques and metrics are discussed. Using machine learning algorithms to develop CF systems is an invaluable asset for both academia and industry and is becoming increasingly crucial for recommending items and services to users.

## Chapter 3: State-of-the-Art Machine Learning Algorithms Applied to Collaborative Filtering

*This chapter is based on:*

A. Althbiti and X. Ma, “Machine Learning,” in *Encyclopedia of Big Data*, L. A. Schintler and C. L. McNeely, Eds. Cham: Springer International Publishing, 2020, pp. 1–5.

### Introduction

Machine learning (ML) is a fast-evolving scientific field that effectively copes with big data explosion and forms a core infrastructure for artificial intelligence (AI) and data science (DS). ML bridges the research fields of computer science and statistics and builds computational algorithms and theories based on statistical models from those fields of studies. These algorithms and models are used by automated systems and computer applications to perform specific tasks, with the aim of high prediction performance and generalization capabilities [22]. ML is also referred to as a predictive analytics or statistical learning. The general workflow of a ML system entails receiving inputs (a.k.a., training sets), training predictive models, performing specific prediction tasks, and eventually generating outputs. Then, the ML system evaluates the performance of predictive models and optimizes the model parameters to obtain better predictions. In practice, ML systems also learn from prior experiences and generate solutions for given problems with specific requirements.

The dramatic evolution of ML as a field of study solves problems that first appeared in the early time of intelligent applications. At that time, systems used hand-coded rules of “if” and “else” decisions to make decisions, for instance, a system that aims to detect fraud transactions. System developers could make up a blacklist of situations that would result in a fraud transaction. Yet, utilizing these hand-coded rule systems to make predictions has significant flaws: (1) the utilized logic to make predictions is specified to a single task, and (2) domain experts who have a deep knowledge of how a prediction should be made are the only ones who design these rules.

The basic failure of these hand-coded rules appears in the facial recognition problem when a set of images is given. However, the use of ML algorithms solves the problem of what characteristics are required to recognize a face. ML algorithms can solve problems that require predictions and automated decision-making. They use a set of training sets as an input, then generate an output. If a training set includes inputs and outputs, the problem will be a supervised learning. There are several examples where supervised learning algorithms are exploited, such as fraud detection in bank transactions, e-mail

filtering, or tumor determining based on a medical scan. On the other hand, if a training set only includes input instances and no target variable, the problem will be an unsupervised learning. Examples of unsupervised learning include, but are not limited to, the identification of topics in a set of corpus, the prediction of ratings in collaborative filtering (CF), or segmentation by an advertising system of users into smaller group based on their preferences and characteristics.

The structure of this chapter is as follows. The present chapter discusses the data representation and feature engineering. Then, it discusses ML algorithms used for CF. Then, it discusses model fitness and evaluation. Finally, it discusses the conclusion.

### **Data Representation and Feature Engineering**

There are three main perceptions about feature representation in ML. The first perception is that the feature engineering means transforming raw data into features (i.e., attributes, concepts, or variables) that better reveal the underlying structure of data. To have a more accurate predictive model, feature engineering should be done carefully and correctly by domain experts. The second perception addresses the feature extraction process, which transforms raw data that has  $m$ -dimensional space into a desired form of  $n$ -dimensional space, where  $m \gg n$ . The last perception determines what attributes are intended to be part of learning a model; this is known as feature selection.

Based on the relationship between attributes, there are two main types of attributes in any scientific experiment to build a predictive model. An independent variable is a variable (often denoted by  $x$ ) being changed or controlled to test effects on a dependent variable. A dependent, responding or target variable is the variable (often denoted by  $y$ ) being tested and measured in a study. Hence, ML provides models that predict a target or dependent variable.

After performing a feature engineering task, ML algorithms are applied based on the data type of a dependent/response variable. The goal is to determine the data type of a dependent variable. There are mainly two data types for a response variable: (1) continuous or (2) categorical/discrete. A continuous feature takes any value between specified ranges of values such as age, while a discrete variable accepts certain number of values such as gender, marital status, or education level. In CF, the provided ratings can be considered categorical and continue based on the developed model. If the model is classification based, then the ratings of the items are categorical, however some researchers consider the problem of CF as a regression problem [40].

### **Machine Learning Algorithms**

According to Jordan and Mitchell, the main paradigms are: (1) supervised learning, (2) unsupervised learning, and (3) reinforcement learning [22]. ML approaches are categorized according to two criteria:

(1) the data type of a dependent variable and (2) the availability of labels of a dependent variable. The former criterion is categorized into two classes: (1) continue and (2) discrete. The latter criterion is utilized to determine the type of ML algorithm. If a dependent variable is given and labeled, it is a supervised learning approach. Otherwise, if a dependent variable is not given or unlabeled, it is an unsupervised learning approach.

Supervised learning algorithms are often utilized for predicting tasks and building a mathematical model of a set of data that includes both inputs and desired outputs. These algorithms learn a prediction model that approximates a function  $f(x)$  to predict an output  $y$  [104]. On the other hand, unsupervised learning algorithms are often used to study and analyze a dataset and learn a model that discovers a useful structure of the inputs without the need of labeled outputs. They are also used to address two major problems in CF: (1) data sparsity, where missing values can affect a model's accuracy and performance, and (2) large dimensionality space, which means data is organized in high-dimensional spaces (e.g., thousands of dimensions).

Mnih et al. discuss that reinforcement learning forms a major ML paradigm that sits at the crossroads of supervised and unsupervised learning [105]. For reinforcement learning, the availability of information in training examples is intermediate between supervised and unsupervised learning. In other words, the training examples provide indications about an output inferred by the correctness of an action. Yet, if an action is not correct, the challenge of finding a correct action endures [22]. Other ML approaches emerge when researchers develop combinations across the three main paradigms, such as semi-supervised learning, discriminative training, active learning, and causal modeling [22].

As a result of using ML algorithms on a training dataset, a model is learned and ready to be used to make predictions on new datasets. Table 3.1 discusses different ML algorithms based on the availability of labeled output variables and their data types. Table 3.2 discusses the state-of-the-art models of ML algorithms used for CF [38].

### *Classification and Regression*

There are several algorithms borrowed from the field of ML to develop models to make a prediction of unseen item in a CF system.

Table 3.1 Classification of machine learning models

	<b>Data are labeled (supervised learning)</b>	<b>Data are unlabeled (unsupervised learning)</b>
<b>Continuous</b>	Regression	Dimensionality Reduction
<b>Discrete</b>	Classification	Clustering

Table 3.2 Different categories of ML models

<b>ML paradigm</b>	<b>Task</b>	<b>Type of algorithm</b>	<b>Model</b>
<b>Supervised</b>	Prediction	Regression and Classification	Linear regression
			Ridge regression
			Least Absolute Shrinkage and Selection Operator (LASSO)
			$k$ -nearest neighbors for regression
			$k$ -nearest neighbors for classification
			(Logistic regression)
			One-vs.-rest linear model for multi-label classification
			Decision Trees (DSs)
			Bayesian classifiers
			Support Vector Machines (SVMs)
Artificial Neural Networks (ANNs)			
<b>Unsupervised</b>	Features extraction	Dimensionality Reduction	Principal Component Analysis (PCA)
			Singular Value Decomposition (SVD)
	Description	Clustering	$k$ -means
			Density-based spatial clustering of application with noise (DBSCAN)
			Message-passing
			Hierarchical
		Association Rule Mining	Apriori

### Linear regression

Linear regression is a statistical model for modeling the relationship between a dependent variable  $y$  and one or more independent variables  $X$ . Equation (3.1) is a linear model for several explanatory variables represented by a hyperplane in higher dimensions:

$$\hat{y} = w[0] * x[0] + w[1] * x[1] + \dots + w[p] * x[p] + b \quad (3.1)$$

where  $x[0]$  to  $x[p]$  signify features of a single instance and  $w$  and  $b$  are learned parameters by minimizing the *mean squared error* between predicted values  $\hat{y}$  and true values of  $y$  on the training set. Linear regression also forms other models such as Ridge and LASSO. Moreover, a regression model, namely logistic regression, can be applied for classification where target values are transformed into two classes as the prediction (3.2) shows:

$$\hat{y} = w[0] * x[0] + w[1] * x[1] + \dots + w[p] * x[p] + b > 0 \quad (3.2)$$

where the threshold of a predicted value is zero. Thus, if a predicted value is greater than zero, a predicted class is +1; otherwise it is -1.

### K-nearest neighbors (KNNs)

KNNs models are used to make a prediction for a new single point (a.k.a., instance). It is used for classification and regression problems and is known as lazy learner because it needs to memorize the training sets to make a new prediction (i.e., instance-based learning). This model makes a prediction for a new instance based on the values of the nearest neighbors. It finds those nearest neighbors by calculating similarities and distances between a single point and its neighbors. The similarity is calculated using Pearson correlation coefficients, cosine similarity, or Euclidian distance.

### Decision Trees (DSs)

DSs classify a target variable in the form of a tree structure. The nodes of a tree can be: (1) decision nodes, where their values are tested to determine to which branch a subtree moves, or (2) leaf nodes, where a class of a data point is determined. Decision nodes must be carefully selected to enhance the accuracy of prediction. DSs can be used in regression and classification applications.

### Bayesian classifiers

Bayesian classifiers are a probabilistic framework for addressing classification and regression needs. It is based on performing Bayes' theorem and the definition of conditional probability. The main assumption of applying Bayes' theorem is that features should maintain strong (naïve) independence.

### Support Vector Machines (SVMs)

SVMs are classifiers that strive to separate data points by finding linear hyperplanes that maximize margins between data points in an input space. It is noteworthy that SVMs can be applied to address regression and classification needs. The support vectors are data points that fit on maximized margins.

### Artificial Neural Networks (ANNs)

ANNs are models inferred from biological neural networks of the brain. They develop a network of inter-connected neurons which work together to perform prediction tasks. Numerical weights are assigned to the links between nodes and are tuned based on experience. The simple representation network consists of three main layers: (1) input layer, (2) hidden layer, and (3) output layer. Handwriting recognition is a typical application that uses ANNs.

### *Dimensionality reduction*

The poor performance of ML algorithms is often caused by the number of dimensions in a data space. Hence, an optimal solution is to reduce the number of dimensions while the maximum amount of information is retained. Principal Component Analysis (PCA) and Singular Value Decomposition (SVD) are the main ML algorithms that offer a solution to the issue of dimensionality.

### *Clustering*

Clustering is a popular ML algorithm which falls in the unsupervised learning category. It groups data points based on their similarity. Thus, data points that fit in one cluster or class are different from the data points in another cluster. A common technique of clustering is a  $k$ -means, where  $k$  indicates total number of clusters. The  $k$ -means clustering algorithm randomly selects  $k$ -number of data points and plots them on a Cartesian plane. These data points form a centroid of each cluster, where the remaining data points are assigned to the best centroid. Then, a process of reassigning centroids is repeated inside each cluster until there are no more changes in a set of  $k$  centroids. Other ML algorithms that consider as an alternative selection of  $k$ -means are DBSCAN, message-passing clustering, and hierarchical clustering.



### *Association rule mining*

Association rule-mining algorithms are mostly used in marketing when predicting co-occurrence of items in a transaction. They are widely utilized to identify co-occurrence relationship patterns in large-scale data points (e.g., items or products).

### **Model Fitness and Evaluation**

The main objective of adopting ML algorithms for training dataset is to generalize a learned model to make accurate predictions on new data points. Hence, if a model makes accurate predictions on new data points, this model would be generalized from a training dataset to test datasets. However, an intensive training of a model increases the complexity. This intensive training may cause the overfitting problem. The overfitting problem means that a model memorizes the training dataset and performs well on training dataset but is not able to make accurate predictions on test datasets. On the other hand, if a model is not sufficiently trained on a training dataset, this model most likely will perform poorly even on a training dataset. Hence, the goal is to select a model that maintains an optimal complexity of training.

Learning a model requires a set of data points as inputs to train a model, a set of data points to tune and optimize a model's parameters, and a set of data points to evaluate its performance. Therefore, a dataset is divided into three sets: the training set, the evaluating set, and the testing set. The method of dividing these sets depends on algorithm developers. There are different techniques to be followed when dividing datasets. One basic technique is to utilize a 90/10 rule of thumb, which means 90% of a dataset is used to learn a model, and the other 10% is used to evaluate and adjust it. Other methods for dataset splitting include  $k$ -fold cross-validation and hold-out cross validation [106]. Furthermore, there are other sophisticated statistical evaluation techniques applicable for different types of datasets, such as bootstrapping methods, which depend on random sampling with replacement, or grid search.

A wide range of evaluation criteria can be used for evaluating ML algorithms. For example, accuracy is an extensively utilized property to assess the performance of model predictions. Typical examples of accuracy measurements are  $R^2$  and Root mean squared error (RMSE). Other metrics for the accuracy of usage prediction include precision, recall, support, and F-score. These evaluation metrics are thoroughly discussed in chapter 2.

### **Conclusion**

ML provides models that learn automatically through experience. The explosion of big data is the main motivation behind the evolution of ML approaches. A survey of the current state of ML algorithms used for CF is introduced in this chapter to simplify the rich and detailed content. This chapter also

demonstrates the use of ML algorithms for CF. To conclude, ML algorithms applied for CF can ultimately address the issues of data sparsity and dimensionality in CF and offer solutions to automate prediction and detection.

## Chapter 4: Addressing Data Sparsity in Collaborative Filtering Based Recommender Systems Using Clustering and Artificial Neural Network

*This chapter is based on:*

A. Althbiti, R. Alshamrani, T. Alghamdi, S. Lee, and X. Ma, “Addressing Data Sparsity in Collaborative Filtering Based Recommender Systems Using Clustering and Artificial Neural Network,”. *Proceedings of the 2021 Computing and Communication workshop and Conference, In Press.*

### Introduction

Collaborative filtering (CF) is fundamentally characterized by recommender systems (RSs), which have recently attracted researchers’ attention [1]. The early evolution of RSs over the last few years is the direct result of the advancements of information retrieval (IR) and information filtering [2]. The main functionality of RSs can be seen as the social process of suggestions and recommendations which ultimately reduce the variety of selections and present more useful selections to users; thus, it was rational to consider RSs as a subfield of IR. Yet, the mid-1990s did witness the birth of RSs as an independent field of study [1, 3]. The ever-increasing data about users and items and the emergence of machine learning approaches have motivated the recent development of RSs. Although the development of RSs technology has been influentially driven by the exceeding use of the web [3], the richness of RSs applications offers researchers many opportunities for building more robust applications [36].

CF can be defined as systems and software tools that automatically and effectively generate recommendations of the most suitable items to a target user by predicting a user’s predilections and preferences [5]. The prediction process relatively depends on the previous knowledge of users’ interests, the description of items, and the interactions between users and items [6]. The objective of developing CF systems is to effectively reduce the overload of available selections by exposing users to the most relevant and suitable items and services from a variety of alternatives.

These systems generate personalized and un-personalized recommendations by developing models that analyze users’ data, opinions and behavior, and also analyze items’ descriptions and attributes to strategically predict a user’s preferences. Yet, they analyze not only a user’s data and opinions but also the opinions of users with similar preferences for augmenting prediction accuracy [109]. The evolution

of the CF arises from a phenomenon in which people seek recommendations from others who share their interests [1, 7, 8].

Using machine learning (ML) algorithms to develop CF systems is extremely valuable for both academia and industry and is becoming increasingly crucial for recommending items and services to users. Different techniques and methodologies employed in CF systems typically follow three steps carried out in succession: data preprocessing, model learning, and the interpretation of results [38]. The particular objective of this empirical study is to address the sparsity problem in CF. The research questions are as follows:

- 1- How could clustering and artificial neural network (ANN) solve the sparsity problem in CF?
- 2- Does the use of different datasets collected from popular domains present a significant difference in the accuracy of recommendation predictions when applying clustering and ANN?

The discussion structure of this chapter is as follows. It first research relevant to the present study. Then, it discusses the proposed model. Then, it discusses the experiments, the results and the evaluation. Finally, it discusses the analysis of the results and the conclusion.

### **Related Research**

A recommendation technique that depends on a rating structure encounters a common problem: sparsity [111, 112]. The data sparsity negatively affects the quality of the recommendations. It arises in several application domains due to user interactions with a small portion of items [110]. For instance, MovieLens dataset includes a rating matrix where users rate movies. The rating matrix is not fully specified; approximately 90% of the matrix has null values. Thus, the traditional CF techniques suffer from the sparsity problem. As a result, it is difficult for CF to generate good recommendations. Previous research has been conducted to solve the sparsity problem. Researchers in [111] use the average of the provided ratings to fill the rating matrix. Other researchers propose models including Matrix Factorization (MF), computational intelligence, and mathematical calculation [113].

MF techniques are introduced to solve the problems of data sparsity and large dimensionality. Several participating teams have used MF, which successfully increases the predictive accuracy when Netflix in 2009 offered a prize for the best CF system that yields better accuracy [89]. MF techniques map users and items into a reduced latent space. The dimensionality reduction can capture high-level patterns in the rating matrix, explain the relationships between users and items, and capture their most hidden characteristics [14, 99]. Researchers in [116] presented a number of MF models for CF that leverage metadata as a bridge between the preferred items by users in different domains to solve the sparsity problem. They claimed that in case the underlying knowledge graph connects items from

different domains, their models can provide better recommendations to new users while keeping a better trade-off between recommendation accuracy and diversity.

Other researchers have tackled the sparsity problem by proposing a model that relies on ML and data mining algorithms. Researchers [113] presented a sparsity alleviation recommendation approach that achieves a better product recommendation performance. In their research, the new sparsity alleviation approach addressed the null or zero values in the rating matrix. The process was done through the use of the multiplication convergence rule and constraint condition. Moreover, another method was proposed to overcome the data sparsity problem by combining MF model with Linked Open Data (MFLOD) [110].

Other approaches that involve mathematical calculations are characterized into probability methods and similarity methods. Researchers have proposed a model that learns domain concept and use a probabilistic method to find similar patterns in the rating matrix [114]. Moreover, a novel method called FRAIPA was designed to solve the sparsity and dynamic data problems, and it ultimately improved the prediction accuracy [115]. Considering the similarity methods, researchers in [117] proposed a novel approach that computes the similarity between users not only based on the items rather the attributes of the items. In their model, users' liking and disliking of the similar characteristics of a particular item were considered separately.

### **The Proposed Model**

This section discusses Clustering and Artificial Neural Network Based Collaborative Filtering (CANNBCF) model that systematically addresses the sparsity problem in CF using: (1) clustering and (2) ANN, as demonstrated in Figure 4.1.

#### Data Preprocessing

The first step is to perform the preprocessing process to prepare these datasets to be used by the proposed algorithm. While examining the main preprocessing steps that real-life data typically requires, this study performs these steps. Researchers in [83] point out that dropping users who rate less than the average rated items per user in the whole dataset may improve the algorithm's performance. It is noteworthy that the problem of cold start [9] occurs when a user rates a few items or has not rated any item. This problem is also applicable to items, as an item may have been rated by few users or not rated at all. Therefore, this study calculates the average number of rated items per user in each dataset and considers only users and items that have the minimum required number of ratings.

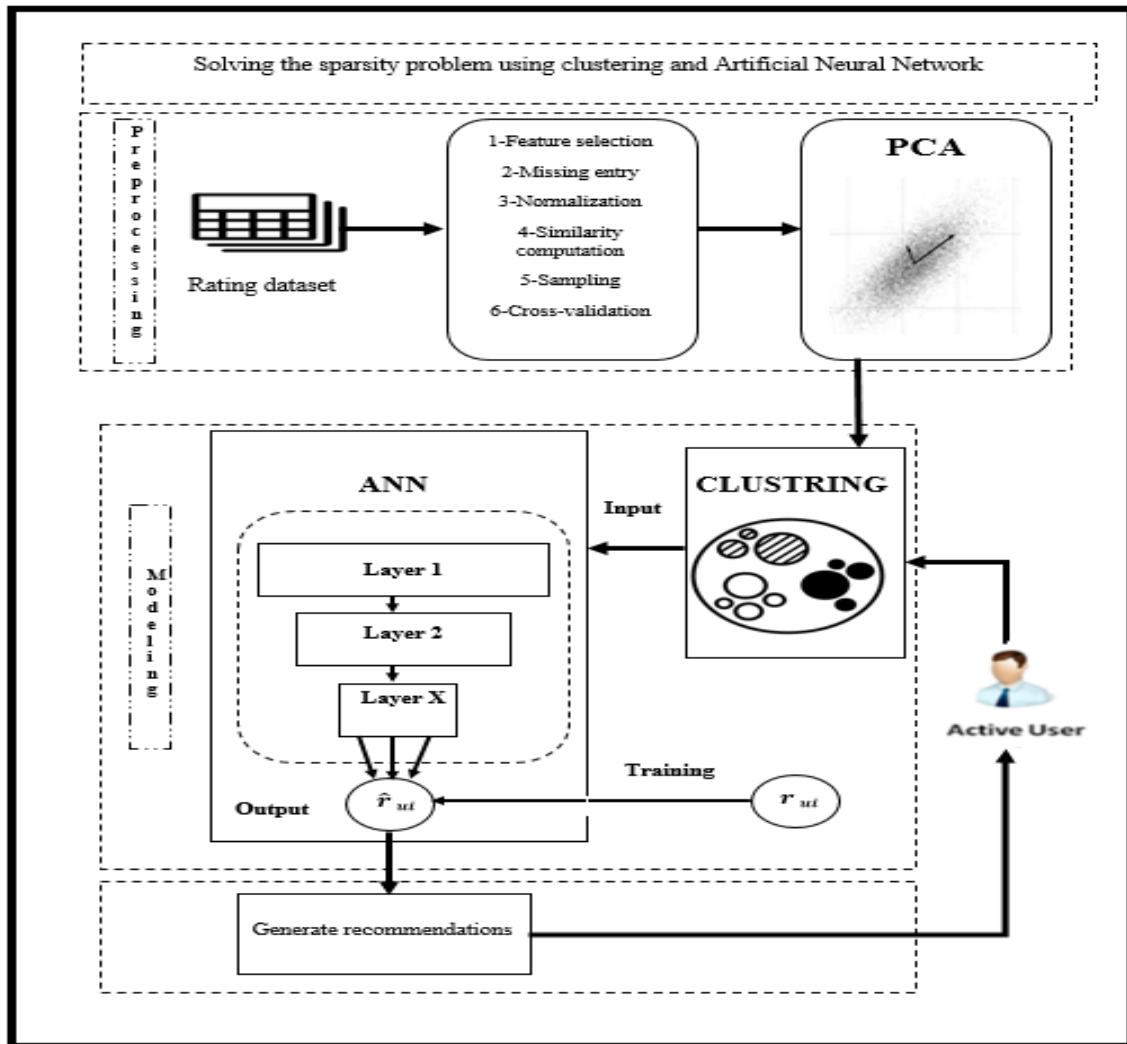


Figure 4.1 The framework architecture of CANNBCF

A missing entry or interaction in the rating datasets is a normal issue as users may rate a few items or items might be rated by a few users. These missing ratings introduce the problem of data sparsity, which is highly correlated with the problem of cold start. The missing entry or interaction in the rating datasets is addressed using the average of the provided ratings to fill the rating matrix.

Another essential preprocessing step is considered when preparing the data. This step ascertains the normal distribution of ratings [9]. Although the distribution of user-item interactions in real-life datasets is normally skewed, several normalization techniques are introduced to uniformly distribute the ratings over the items. The normalization step also helps to map users' ratings to a representative universal scale, since the rating process depends on each user's personal scale. One common normalization scheme is often used to identify the polarity of a rating whether it is positive, neutral, or negative. It is known as

mean-centric. Yet, although the normalization of ratings might improve the performance of the algorithm, there is a chance that it will introduce undesirable effects in some cases.

Moreover, calculating the similarity between users and items is a crucial step for generating predictions and recommend items. It can have a positive effect on both the accuracy and performance of CF [9]. Computing the similarity allows the ratings of trusted neighbors to be used while predicting the ratings for unseen items to a similar user. It also helps to determine the effect of the neighbors' ratings. These algorithms highly depend on selecting an appropriate distance measure that computes the similarity weights. The first common approach to compute the similarity weights is Cosine Vector. It measures the similarity between two instances after representing them in the form of a vector  $x_a$  and  $x_b$ . Another common measure that depends on the effects of the mean and the variance of the ratings is known as Pearson Correlation coefficient.

Concerning the sampling size to train and test the proposed algorithm, this study uses a common technique  $K$ -fold cross-validation to avoid the overfitting problem (a.k.a., over-specialization problem). The process is done for  $k$  times until each group is treated as validation, while the remaining is treated as training data. Hence, 5-fold cross-validation is used where the average performance of the  $K$ -learned models is calculated.

The last step of the preprocessing process is dimensionality reduction. The most representative method is the Principal Component Analysis (PCA) [38]. It is used for the feature extraction process, where an  $m$ - dimensional space is reduced into an  $n$ -dimensional space. The reduction process is done while most of the information is retained and kept. Table 4.1 discusses the fundamental preprocessing steps.

Table 4.1 The description of the preprocessing steps

<b>Data preprocessing steps</b>
1-Number of ratings per user/item $\geq$ the number of the average total rating per user /item to avoid the cold start problem
2-Missing entry is estimated to be equal to the mean of corresponding row/column in the matrix
3- Normalization is done through mean-centric technique to transform ratings to a universal scale
4-The method to compute the similarity weights is Cosine Vector
5-Sampling is done through using standard random sampling without replacement with an 80/20 proportion $k$ -fold to avoid the overfitting problem
6- Dimensionality reduction technique: PCA

### Clustering

The unsupervised machine learning algorithm used first is clustering. The main problem of developing a CF algorithm is the number of operations needed to compute distances between neighbors to find the best k-nearest neighbors. The clustering algorithms can play a vital role to compute the distance between two objects. Researchers claim that clustering is used to improve efficiency because the number of operations is reduced [38]. Clustering assigns items to groups so that items in the same group are more similar than items in other groups. Minimizing intra-cluster distances while maximizing inter-cluster distances is the main goal of a clustering algorithm. The k-means clustering algorithm is used because it is an extremely efficient algorithm [38]. In particular, the used clustering technique is the locality-sensitive hashing (LSH). LSH hashes similar input items into the same "buckets" with high probability [107].

### Artificial Neural Network

The second supervised machine learning algorithm that is used is ANN. It is characterized as forecasting algorithms that predict the future ratings for unseen items based on the previously recorded patterns [5,118]. ANN learns a model by composing layers that perform functions to predict ratings for unseen items. It is noteworthy that ANN can have any number of layers. They are input, hidden, and output layers. Also, there are different functions that interconnect these layers. For instance, the simplest implementation of ANN is the perceptron model, which has two functions known as Threshold and Summing. This demonstrates the simplest case of ANN that trains a model with sufficient data to obtain the required rating prediction.

The proposed neural network architecture comprises three inputs: (1) user features, (2) item features, and (3) similar user features, as demonstrated by Figure. 4.2. It passes each input to a separate fully connected layer and batch normalization Rectifier (ReLU). Moreover, it takes the average of the embedding obtained for the nearest neighbor user to obtain a single representation for the similar users to a target user. It then concatenates the three representations and passes them through three layers of fully connected layers, two of which have ReLU as their activation function, and the third of which the output of the last layer is the predicted rating.



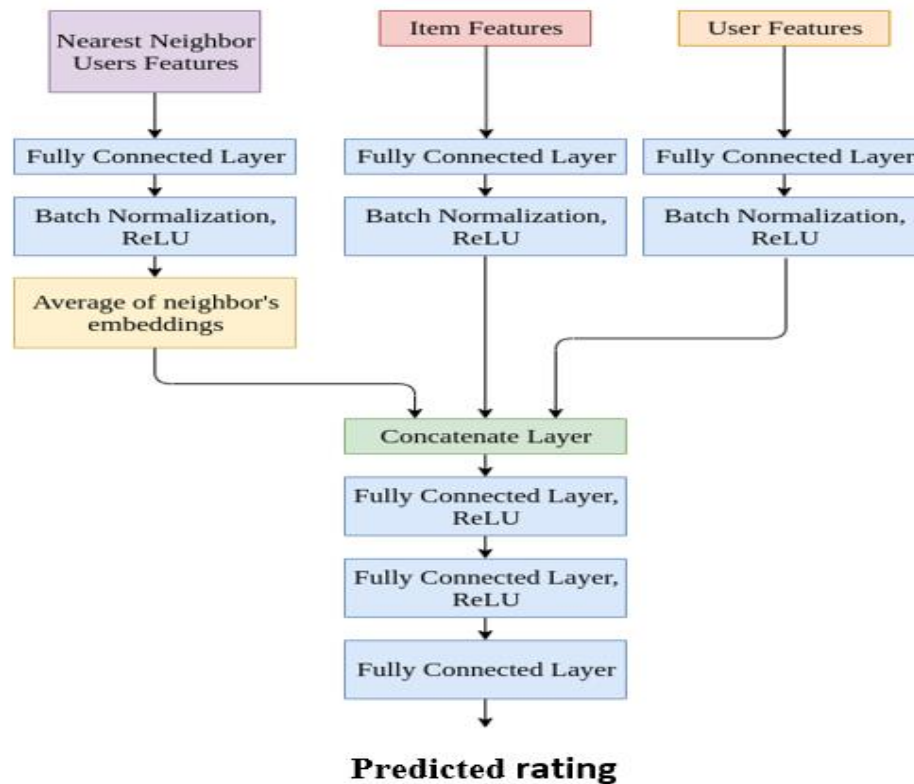


Figure 4.2 The architecture of the ANN used to predict ratings

## Experiments

This section discusses the used datasets, setup of the experiments, and the evaluation.

### Datasets

To address the proposed research questions, this study conducts a set of experiments. It identifies the most used datasets from several research papers [39,83,108]. The researchers survey the publicly available recommendation datasets, which provide researchers a way to evaluate the performance of their proposed algorithms. The selection of these domains is motivated by [39], since the researchers claim that the most used datasets are from these domains. Table 4.2 demonstrates the description and the main characteristics of the used datasets to evaluate the role of the proposed algorithms.

Table 4.2 The description of the used datasets

Data	Jester	MovieLens 1 M	Book-Crossing	Last.fm
No. of ratings	4.1 million	1 million	1.1 million	92,834
No. of users	73,496	6,040	278,858	1,892
No. of items	100	3,592	271,379	17,632
Range	-10 to +10	1,2, ..., 5	1,2, ... , 10	1,2, ..., 5
Domain	Joke	Movie	Book	Music

These datasets are “Jester,” “MovieLens,” “Book-Crossing” and “Last.fm”. The availability of these datasets on the Web and the typical use by developers and researchers in CF techniques motivate the selection. Therefore, this study uses these datasets to evaluate the proposed algorithms.

(1) Jester Dataset

The Jester is a WWW-based joke RS. The Jester dataset contains 4.1 million ratings entered by 73,496 users for 100 jokes. User ratings are explicitly recorded on a real value, ranging from -10 to +10.

(2) MovieLens 1 million

The MovieLens dataset is a movie RS. It has 1.1 million ratings entered by 6,040 users for 3,592 movies. User ratings are explicitly recorded on a real value, ranging from 1 to 5.

(3) Book-Crossing

The Book-Crossing dataset is a book RS. It has 1 million ratings entered by 278,858 users for 271,379 books. User ratings are explicitly recorded on a real value, ranging from 1 to 5.

(4) Last-fm

The Last-fm is a popular music platform. It contains 92,834 ratings entered by 1,892 users for 17,632 items. User ratings are implicitly inferred by mapping listening counts into real values of 1 to 5.

Equation (4.1) is defined in [83] as follows:

$$r = \begin{cases} \lfloor \log_{10} l \rfloor + 1, & \text{if } \lfloor \log_{10} l \rfloor + 1 \leq 5 \\ 5, & \text{otherwise} \end{cases} \quad (4.1)$$

where  $l$  is the listening count,  $r$  is the implicit rating after transforming  $[\cdot]$  is the rounding operator towards zero.

### Experiments Setup

The codes are implemented using PyTorch library for the neural network training and inference. Also, Faiss library is used for fast nearest neighbor search. The classes are not balanced; therefore, the distribution of the number of samples per each class is not uniform. The imbalance will lead to a biased prediction because during the learning, the classifier gets biased to the highly repeated classes. As a result, weighted cross-entropy is used as the loss function, where the weights are proportional to the inversion of the class frequencies. The training samples are the triples of (userID, itemID, rating) mapped into (userFeatures, userNeighborFeatures, itemFeatures, rating). The userFeatures and itemFeatures are obtained from the PCA output, and userNeighborFeatures are obtained by finding the nearest neighbors of the user. The rating predictor is trained for 150 epochs using Adam optimizer with the learning rate of  $e^{-4}$ .

### Evaluation

The evaluation process is done based on historical datasets. This type of evaluation is known as offline evaluation, in which the data are previously collected. This research aims to evaluate the proposed method over a range of evaluation criteria widely used when deciding which algorithm to use. These evaluation criteria include Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), accuracy, precision, recall, the area under the receiver operating characteristic (AUROC), and F-score. They are used to evaluate the accuracy of usage predictions and accuracy of the ratings predictions. The process of this evaluation is performed using datasets that consisting of users and their consumed or preferred items. Then, a ranking accuracy of a CF is evaluated after hiding some of a test user's selections.

- (1) Accuracy is the most intuitive performance measure and is estimated as a ratio of correctly predicted observations to the total observations, given by (4.2),

$$Accuracy = \frac{tp+tn}{tp+fp+fn+tn} \quad (4.2)$$

- (2) Precision (a.k.a., positive predictive value) is a ratio of the number of true positive recommended items to all the recommended items, given by (4.3),

$$Precision = \frac{tp}{tp+fp} \quad (4.3)$$

- (3) Recall (a.k.a., true positive rate, sensitivity) is a ratio of the number of true positive recommended items to all the consumed items, given by (4.4),

$$Recall = \frac{tp}{tp+fn} \quad (4.4)$$

- (4) AUROC curve is a useful measurement for comparing several algorithms independently of application [5]. It is a graph showing the performance of a classification model at all classification thresholds. The receiver operating characteristic (ROC) is a probability curve, and an area under the curve (AUC) represents the degree or measure of separability. It tells how much a model is capable of distinguishing between classes.

- (5) F1-score is the harmonic mean between the precision and the recall and reveals a better quantification than either the precision or the recall [3], given in (4.5),

$$F1 = \frac{2*precision*recall}{precision+recall} \quad (4.5)$$

- (6) Root mean squared error (RMSE) is a popular metric, given by (4.6),

$$RMSE = \sqrt{\frac{1}{|\mathfrak{S}|} \sum_{(u,i) \in \mathfrak{S}} (\hat{r}_{ui} - r_{ui})^2} \quad (4.6)$$

- (7) Mean absolute error (MAE) is another alternative metric given by (4.7),

$$MAE = \frac{1}{|\mathfrak{S}|} \sum_{(u,i) \in \mathfrak{S}} |\hat{r}_{ui} - r_{ui}| \quad (4.7)$$

## Results

While conducting the intensive experiments, the accuracy, MAE, runtime, precision, recall, AUROC, and F1-score of the baseline and proposed models are respectively computed on the previously discussed datasets. Table 4.3 shows the average evaluation criteria of NBCF and CANNBCF algorithms on four datasets. The bold font is used to indicate the improved results. It is observed from Table 4.3 that the proposed model CANNBCF significantly achieves better performance and provides more accurate prediction than NBCF. It demonstrates the effectiveness of using ANN to make predictions. Moreover, it is observed that CANNBCF obtains lower MAE than NBCF, which validates the strength of using clustering and ANN. While CNNBCF obtains better accuracy, precision, recall, AUROC, and F1 score than NBCF, these improved results demonstrate the effectiveness of the proposed model. The results

also indicate that the average execute time to make recommendations when using CNNBCF model is lower than the execute time when using NBCF model.

While the high dimensionality on Book-crossing, last.fm, and Movielens datasets is reduced using PCA, these datasets gain large speed up. Therefore, the runtime of CANNBCF is better for Book-crossing, last.fm, and Movielens because the number of items in these datasets is relatively high. On the other hand, the reason for achieving worse runtime for Jester is the fact that Jester has low number of items (100). The performance comparison of NBCF and the CANNBCF algorithms on ROC for the four datasets is depicted in Figure 4.3.

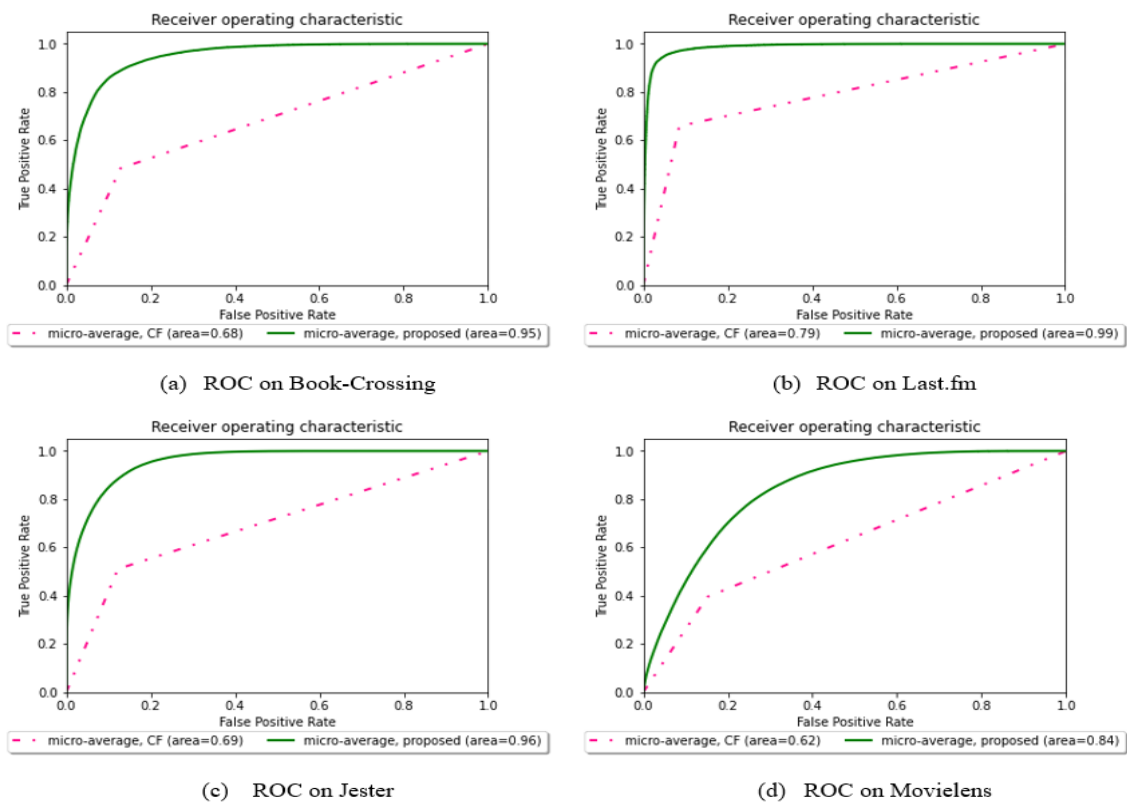


Figure 4.3 Performance comparison of NBCF and CANNBCF on ROC for four datasets

Table 4.3 The averages of the evaluation criteria of NBCF and CANNBCF models on four datasets

Dataset	NBCF							CANNBCF						
	Accuracy	MAE	Runtime (Second)	Precision	Recall	AUROC	F1 Score	Accuracy	MAE	Runtime (Second)	Precision	Recall	AUROC	F1 Score
Book-crossing	0.53	0.57	194.0	0.50	0.48	0.59	0.40	<b>0.75</b>	<b>0.26</b>	<b>2.7</b>	<b>0.73</b>	<b>0.73</b>	<b>0.91</b>	<b>0.73</b>
Jester	0.56	0.53	<b>6</b>	0.57	0.51	0.64	0.43	<b>0.80</b>	<b>0.27</b>	7.4	<b>0.76</b>	<b>0.75</b>	<b>0.94</b>	<b>0.75</b>
Last.fm	0.63	0.34	2.2	0.65	0.66	0.61	0.61	<b>0.93</b>	<b>0.11</b>	<b>0.7</b>	<b>0.89</b>	<b>0.89</b>	<b>0.96</b>	<b>0.89</b>
Movielens	0.51	0.73	9.9	0.46	0.40	0.54	0.27	<b>0.71</b>	<b>0.56</b>	<b>4.7</b>	<b>0.53</b>	<b>0.53</b>	<b>0.79</b>	<b>0.53</b>

The performance comparison of NBCF and the CANNBCF algorithms on Precision-Recall for the four datasets is depicted in Figure 4.4.

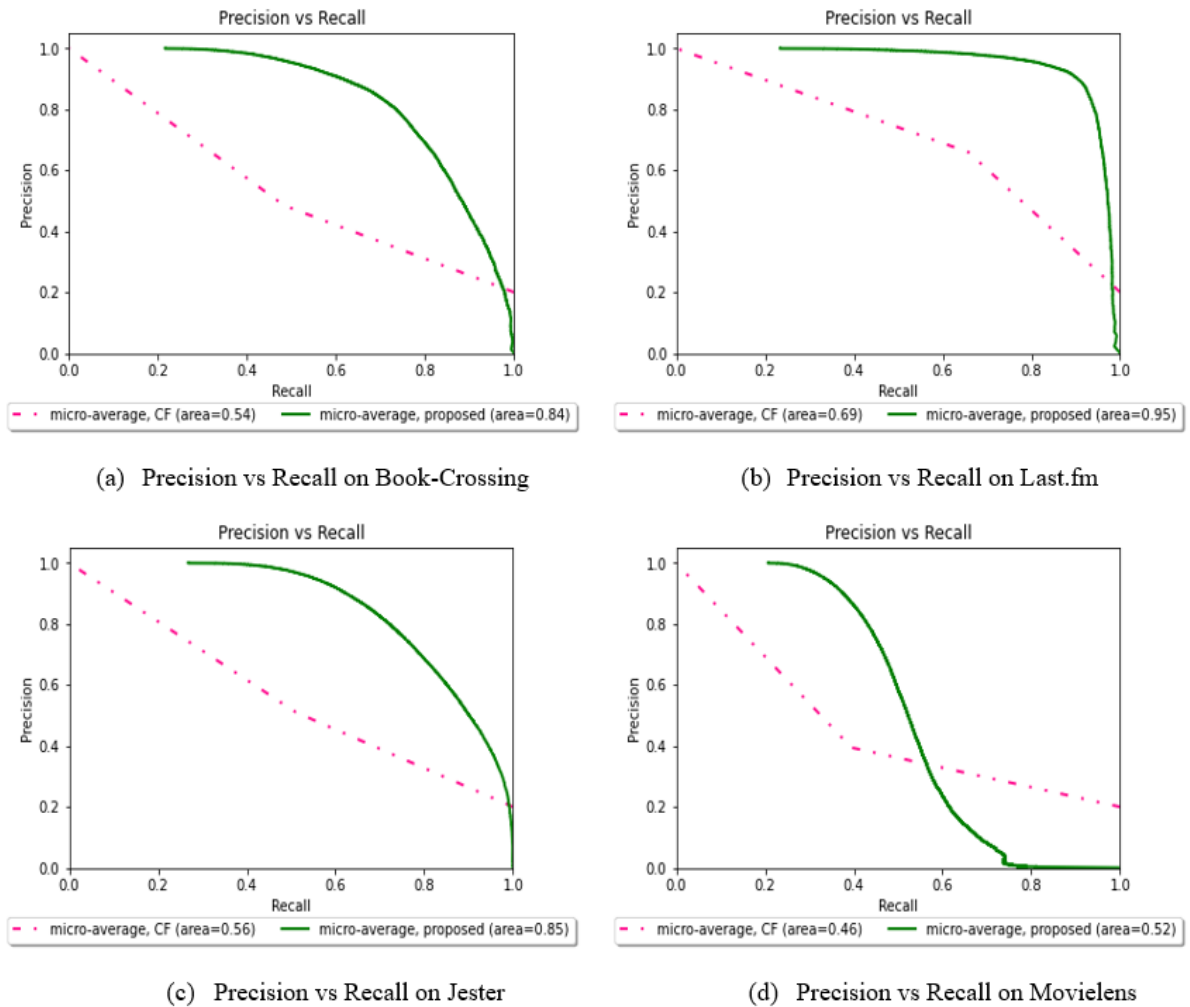


Figure 4.4 Performance comparison of NBCF and CANNBCF on Precision vs Recall for four datasets

The macro-average and micro-average ROC curves are also shown in Figure 4.5, Figure 5.6, Figure 4.7, and Figure 4.8. In macro-average, the AUROC is independently computed for each class and then the average is taken to equally consider all classes. However, in micro-average, the class frequencies are aggregated to compute the average AUROC. For the NBCF, the AUROC of each class is highly correlated with the class frequency. For instance, in the Book-crossing dataset, rating 4 and 3 have the highest class frequency, and their corresponding AUROC is the highest. The least frequent classes often have lower AUROC and area under precision-recall scores. The AUC of macro-average ROC is lower than the AUC of micro-average ROC in all cases. The reason is that the classes that the model is underperforming often have a small amount of training data, so their corresponding class frequency is

low. As a result, when their class frequency contribution is considered equal to the other classes, it adversely reduces the macro-average score. The areas under ROC and Precision-Recall curves are both improved for all the five classes by using CANNBCF compared to the NBCF.

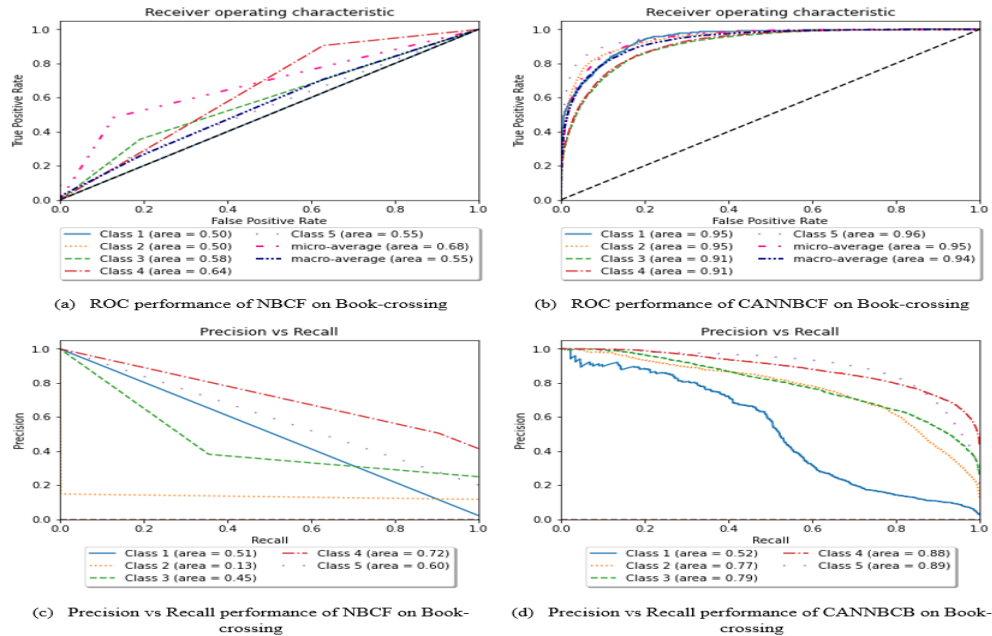


Figure 4.5 Performance comparison of NBCF and CANNBCF on ROC and Precision vs Recall for Book-Crossing

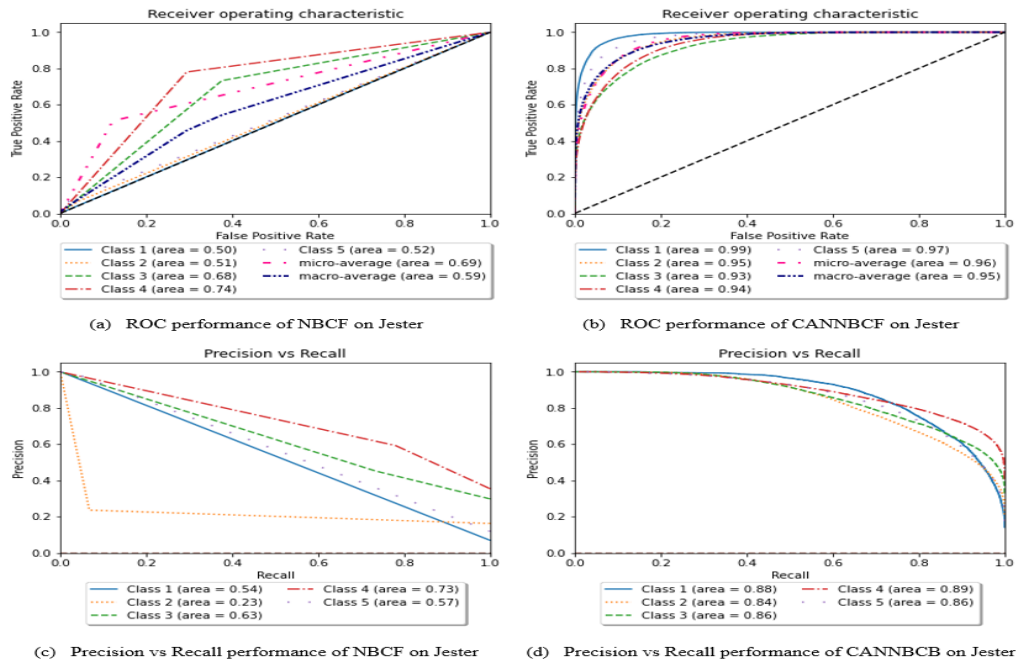


Figure 4.6 Performance comparison of NBCF and CANNBCF on ROC and Precision vs Recall for Jester



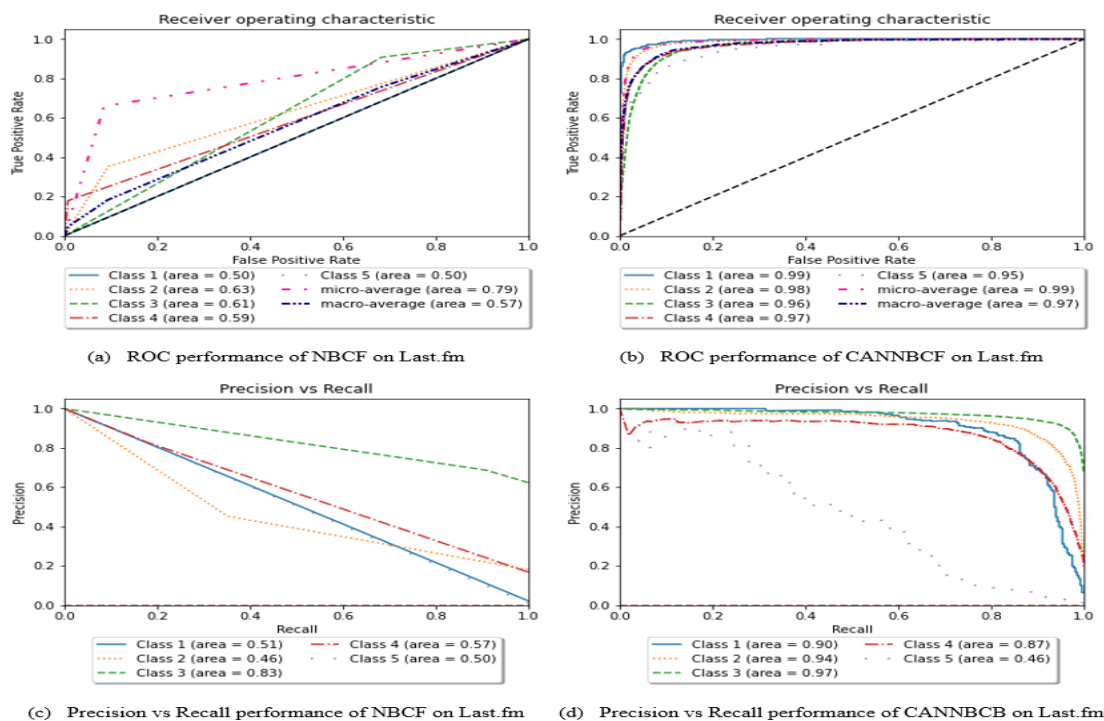


Figure 4.7 Performance comparison of NBCF and CANNBCF on ROC and Precision vs Recall for Last.fm

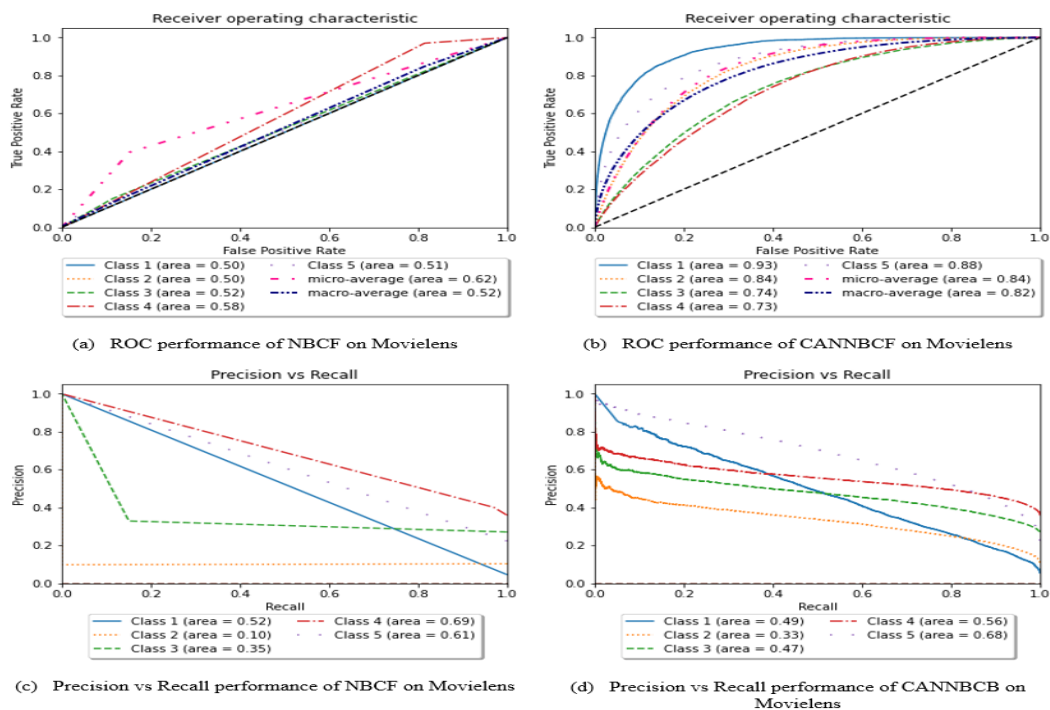


Figure 4.8 Performance comparison of NBCF and CANNBCF on ROC and Precision vs Recall for Movielens

### Discussion and Results Analysis

The use of an ANN allows learning the higher degrees of statistical inter-dependence between user-user and user-item features. The co-relation of the features captured by the neural network adds higher-order information as opposed to the NBCF that uses first-order statistics such as averaging. In addition, a dimensionality reduction algorithm is applied before feeding the data into the neural network, which further alleviates the sparsity problem. The proposed model outperforms the rating prediction accuracy of the NBCF algorithm by 24% on average. The accuracy increase for Book-Crossing, Jester, Last.fm, and MovieLens datasets are 22%, 24%, 30%, and 20%, respectively. Figure 4.9 demonstrates the performance of NBCF and CANNBCF on accuracy for the different datasets. Also, Figure 4.10 demonstrates the performance of NBCF and CANNBCF on RMSE for the different datasets.

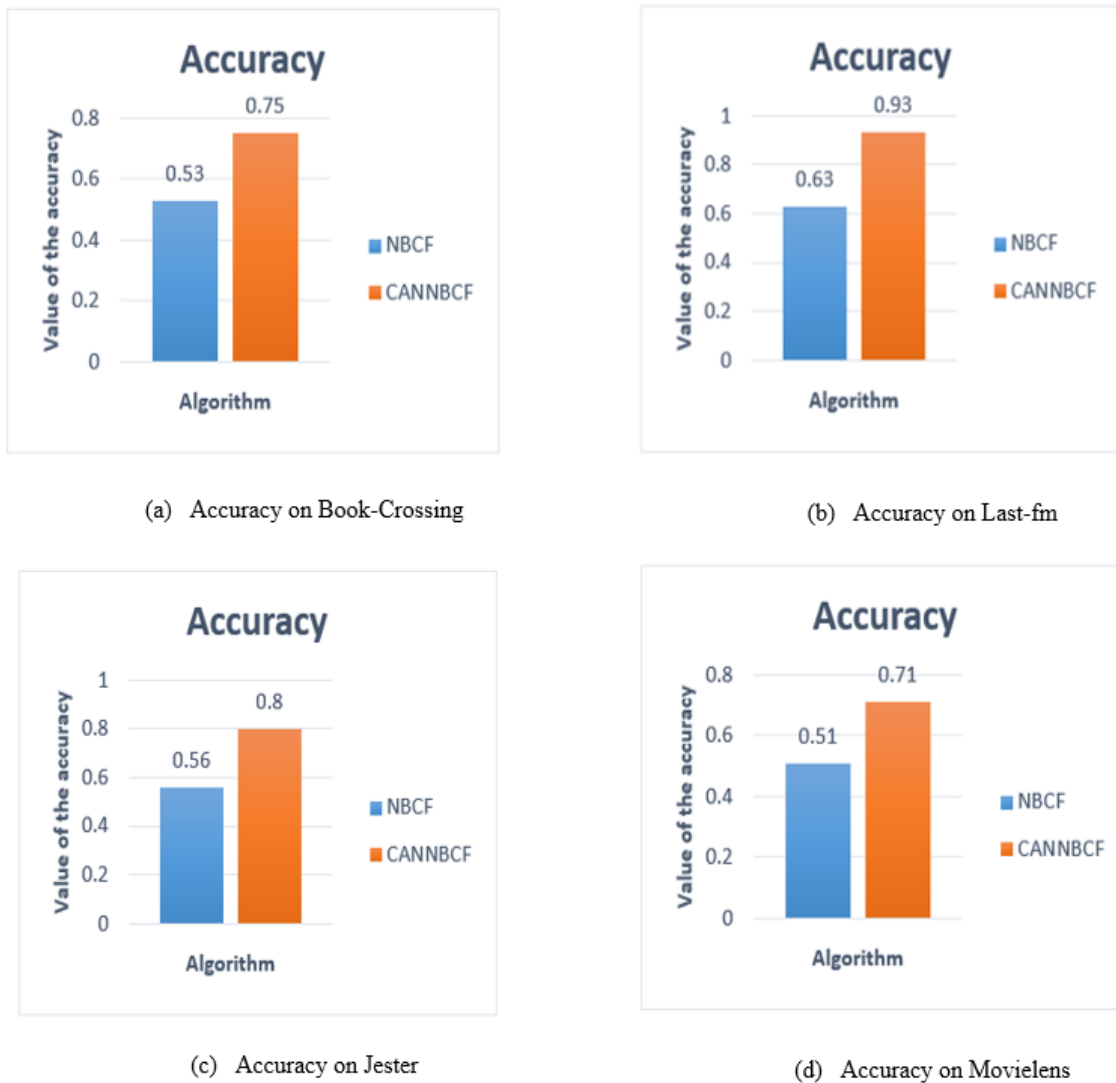


Figure 4.9 Performance of NBCF and CANNBCF on Accuracy for the four datasets

Figure 4.9 and Figure 4.10 show that CANNBCF provides more accurate predictions than NBCF. It validates that CANNBCF model provides a significant improvement in accuracy for the Last.fm dataset and an important improvement in accuracy for Jester dataset with respect to other datasets. Furthermore, Figure 4.10 shows that CNNBCF can lead to better results when more  $k$ -neighbors are considered.

The datasets are sorted based on the value of the accuracy from high to low and their characteristics are discussed accordingly:

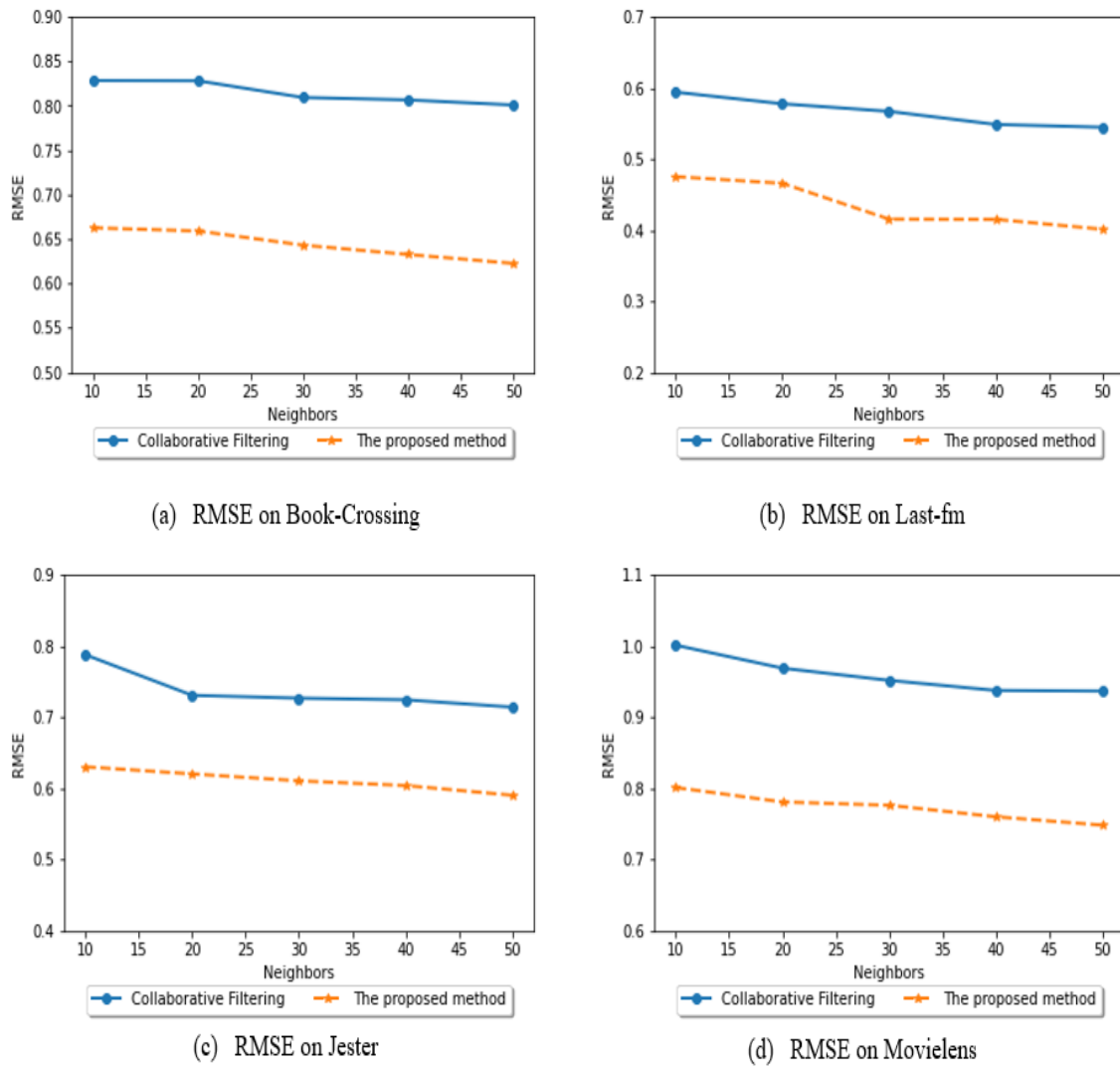


Figure 4.10 Performance of NBCF and CANNBCF on RMSE@K for the four datasets

- (1) Last.fm includes 1826 users and 2384 items after the preprocessing steps. The sparsity level after the preprocessing steps is 98%. It has the highest imbalance ratio where 78% of the ratings are 3, which leads to higher accuracy compared to the other datasets.
- (2) Jester includes 13043 users and 47 items after the preprocessing steps. The sparsity level of this dataset is reduced from 56% to zero as most users were required to rate most jokes, and after filtering, those items and users who rated less than average are dropped. The ratio of users to items is relatively high, which helps to learn a more accurate model for this dataset compared to the Book-crossing and Movielens datasets.
- (3) Book-crossing includes 10612 users and 21507 items after the preprocessing; therefore, the user to items ratio is 0.49. The ratio is relatively smaller than Last.fm and Jester, which makes it hard to predict the ratings accurately. In addition, the class frequency distribution is much more balanced compared to the Last.fm, which makes the prediction task more challenging. There is a 75% sparsity level, and the entropy of the class frequency distributions are 0.77 and 1.36 for Last.fm and Book-crossing datasets, respectively.
- (4) Movielens includes 1887 users and 1212 items after the preprocessing steps. Similar to the Book-crossing dataset, it is harder to learn a model compared to Last.fm and Jester datasets. In Movielens, there is a 75% sparsity level. The entropy of the class frequency distribution for Movielens is 1.42, which is considered the highest among all other datasets. Thus, it is hard to predict the correct rating for all five classes. Figure 4.11 demonstrates the rating distribution of the four datasets.

### **Conclusion**

This chapter addresses the sparsity problem and proposes a model that improves the accuracy of the recommendation in CF. It proposes a hybrid model using ANN and clustering algorithms. CANNBCF shows its ability to reveal better results compared to NBCF. The results reveal that the sparsity problem can significantly affect the accuracy, quality and performance of the conventional CF models used to predict ratings of an active user for inexperienced items. The proposed model shows that accuracy, precision, recall, F-score, and ROC are improved when using a dimensionality reduction algorithm before feeding the data into the neural network, which further alleviates the sparsity problem of CF.

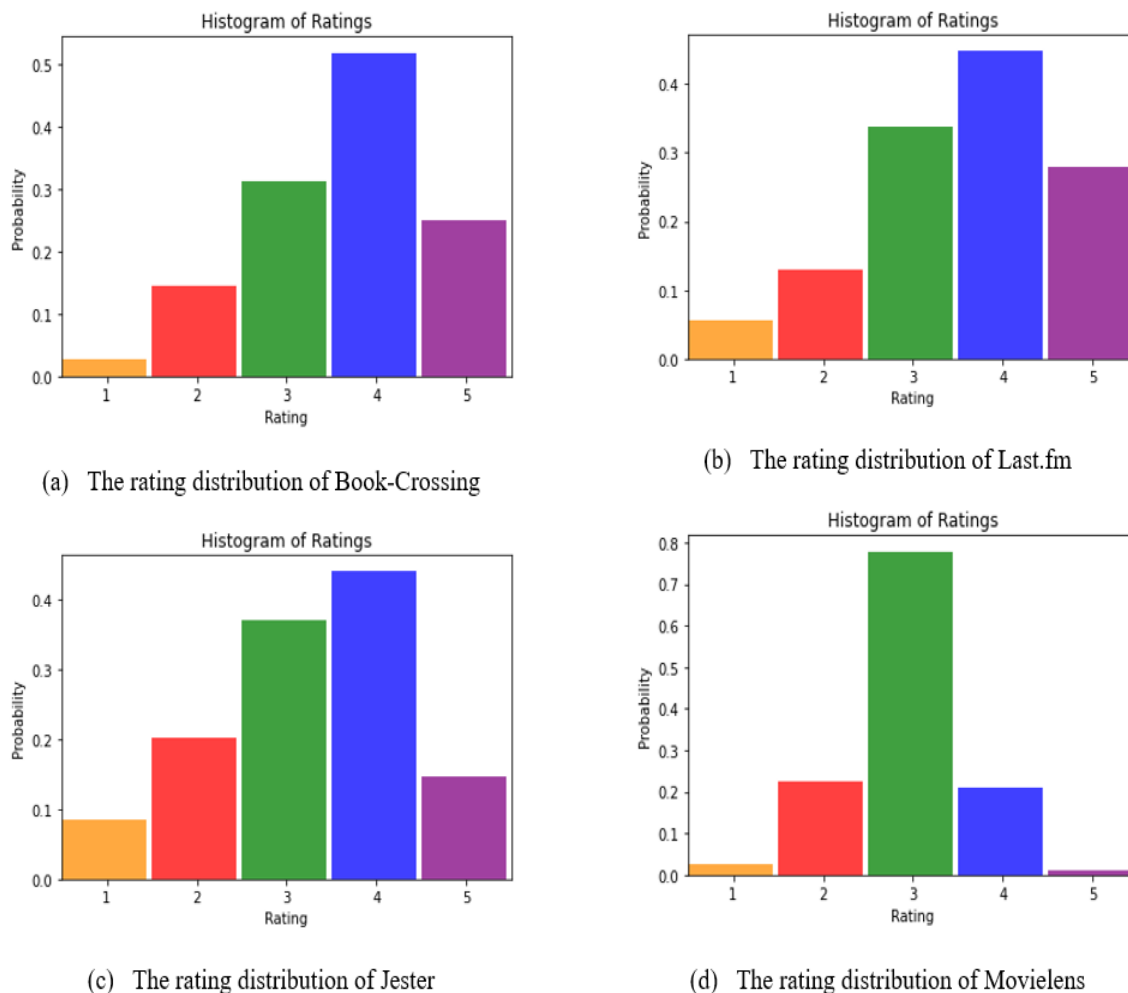


Figure 4.11 Rating distribution of the datasets

The effectiveness of CANNBCF is experimentally examined using four different datasets collected from popular domains. “Jester,” “MovieLens,” “Book-Crossing” and “Last.fm” collected from jokes, movies, books, and music domains, respectively. Intensive experiments are conducted on four datasets while examining the performance of the two algorithms NBCF and CANNBCF. The experiments indicate that the proposed algorithm can solve the sparsity problem and attain significantly better recommendation quality than NBCF. Even though the classes of the datasets are imbalanced, the proposed algorithm is not biased towards the frequent classes.

## Chapter 5: An Efficient Similarity Measure Based on Rating Alignment to Augment Collaborative Filtering

*This chapter is based on:*

**A. Althbiti**, R. Alshamrani, T. Alghamdi, H. Ghanem, and X. Ma, “An Efficient Similarity Measure Based on Rating Alignment to Augment Collaborative Filtering ,”.

### **Introduction**

Collaborative filtering (CF) can be defined as systems and software tools that automatically and effectively generate a list of recommendations of the most suitable items to a target user by predicting a user’s interests and preferences [6]. The prediction process depends on the previous collected knowledge or information about user’s interests, a description of items, and the interactions between users and items [6]. CF systems are commonly used in the field of e-commerce and play a significant role in guiding users to make better decisions to enhance customer relationship management. These systems generate personalized and un-personalized recommendations by developing models that analyze users’ data, opinions, and behavior. These systems also analyze items’ description to systematically predict a user’s preferences. Yet, they analyze not only a user’s data but also the opinions of users with similar preferences to augment the prediction accuracy.

With the rapid growth of machine learning (ML) algorithms, several CF applications have been introduced in several domains to recommend items and services to users-- for instance, e-commerce, e-learning, e-library, e-business, e-tourism, e-government and e-resource services domains [36, 37]. As a field of study, the dramatic evolution of ML algorithms and similarity models addresses drawbacks that appear in the early time of CF systems. Researchers claim that the main research focus of current CF research, especially in the big data era, is how to effectively propose learning and similarity models to overcome data sparsity and limited coverage problems in CF. The limited coverage means the process of finding neighbors is based on the rating of common items (co-rating). The data sparsity in the rating matrix means not all the ratings are provided.

The key component of a CF system is the similarity function. To compute the similarity values between users and items in CF, a similarity function must be carefully chosen. The most commonly used similarity metrics are cosine (COS), Pearson Correlation Coefficient (COR), weighted Pearson Correlation Coefficient (WCOR), Constrained Pearson’s Correlation (CPC), Mean Squared Difference (MSD), Jaccard, and Jaccard Mean Squared Difference (JMSD) [121]. These conventional measures can effectively indicate the degree of similarity between a pair of users or items. It is noteworthy that

the core of these measures is simple and performed effectively when dealing with large number of collaboratively rated (co-rated) items.

These conventional measures can negatively affect the accuracy of the recommendation and the performance of a CF system when the data sparsity and limited coverage problems exist. Moreover, they may have some inadequacies in identifying appropriate  $k$ -nearest neighbors ( $k$ NNs) while dealing with fewer number of co-rated items [119]. To address these problems, several similarity models have been introduced in recent years, though the improvement is not obvious [120]. Therefore, this research proposes an efficient similarity model that strives to identify regions of similarity between users or items in the rating matrix. Based on these identified similar regions, CF can compute the similarity scores between a pair of users or items to predict an active user's ratings on unseen items. Moreover, there are two main advantages of the proposed similarity model. First, the prediction process is not sensitive to the number of  $k$ NNs. The second is that similarity computation does not only consider collaboratively rated (co-rated) items to compute the similarity metrics in relatively sparse datasets, but it also considers aligning the rating vectors of users to classify relevant neighborhoods and perform well in recommendation generation.

The research questions that this empirical study addresses are:

- 1- How could the use of rating alignment-based similarity model address the limited coverage issue in CF?
- 2- Does the use of different datasets collected from popular domains present a significant improvement when using rating alignment-based similarity model?

The other sections of this chapter are organized as follows. It first reviews the literature and several related similarity calculation models and their limitations in the field of CF. Then, it thoroughly discusses the proposed similarity model. Furthermore, it discusses the intensive experiments and conducted on four different datasets and further analyzes the results. Moreover, it discusses the implications and the advantages of the proposed similarity model. After discussing the implications of the proposed model, conclusion is discussed in the last section.

### **Related Research**

CF technique is widely used in the process of recommendation generation. The main components of CF are: (1) the stored data, (2) similarity or correlation computation (e.g., COR), (3) prediction algorithms (e.g., ML algorithms), and (4) the filtered results (final recommendations). The stored data are about: (1) users (e.g., user's demographic factors, or ratings for previous selected items) and (2)

items (e.g., item's descriptions, or ratings provided by users). CF mainly is classified into four techniques: (1) neighborhood-based CF (NBCF), (2) model-based CF (MBCF), (3) graph-based CF (GBCF) and (4) hybrid-based CF (HBCF).

CF predicts arbitrary rating values because of the stochastic prediction practice, dynamic rating data, and subjectivity users. Consequently, an accurate rating prediction is an insignificant methodology in the field of RS. It is noteworthy that researchers are attempting to enhance the performance of CF in terms of evaluation metrics and in particular accuracy metrics.

### *Collaborative Filtering Techniques*

- (1) NBCF technique uses stored user-item ratings directly to generate a list of recommended items. The process of rating predictions is slightly slow because each time a prediction is estimated, NBCF has to access stored ratings, perform calculations and make predictions.
- (2) MBCF technique uses ML algorithms and data mining (DM) models in the context of predictive models. The user-item rating matrix is used to learn a parametric model (a.k.a., latent factor model) that explains relationships between users, items, or both. To learn a predictive model that predicts future ratings for new items, a model should have a set of parameters that capture salient features of users and items. These parameters are learned from training data and exploited to predict ratings [10].
- (3) GBCF technique can address the problem of data sparsity in the user-item rating matrix. Using a structural transitivity while developing graph models can define a similarity in NBCF techniques [3]. These graphs provide structured representations to project the relationships between a pair of users or items.
- (4) HBCF technique can be a combination of all the previous mentioned techniques of CF. Researchers propose a model that address the problem of predicting users' social influences on upcoming events in Event-Based Social Networks (EBSNs) [78]. To solve this problem, researchers introduce a HBCF model, namely, Matrix Factorization with Event-User Neighborhood (MF-EUN), by fusing both event-based and neighborhood methods into a matrix factorization.

### *Neighborhood-Based CF*

The components of NBCF are: (1) ratings' normalization, (2) similarity computation (a.k.a., affinity or correlation computation), and (3) the process of neighbors' selection. The aim of rating normalization process is to transform users' ratings to a general and unified scale [9]. The most common rating normalization techniques are mean-centric and Z-score. The quality of a recommendation system and



the accuracy of a prediction are relatively based on the selection of a similarity function. Several available similarity functions are discussed in [121].

The number of  $k$ NNs plays a vital role in the performance of a CF system. It is not rational to include ratings of every user in the process of predicting rating for an item of an active user. There are principles to follow when selecting the number of  $k$ NNs. Researchers argue that filtering the neighbors should be carefully performed through two main filtering steps: (1) a global filtering step that keeps similar users or items, and (2) a pre-prediction step that only keeps best candidates to be used in the prediction process. It is noteworthy that the optimal number of  $k$ NNs can range from (20 to 50) [43]. However, other researchers argue that the optimal value of  $k$ NNs should be determined by cross-validation function [9].

NBCF can be categorized into two techniques: (1) user-based NBCF and (2) item-based NBCF. User-based technique uses previous collected ratings of users who share the preferences of a specific user to predict ratings for unseen items of the same specific user. It initially finds similar rating patterns among users and a specific user. Then, it computes the similarity and retrieves  $k$ NNs of a specific user. Finally, it uses neighbors' ratings to predict a rating for an unseen item of a specific user.

On the other hand, item-based technique uses previous stored ratings in the user-item rating matrix to predict ratings for unseen or unexperienced items of the same user. It does not consider similar users' taste; instead, it completely considers provided ratings of a specific user to a list of items. These items share similarities as the unseen items for a specific user. Hence, the actual ratings of a specific user are used to predict rating for a target item. It is noteworthy that some users prefer item-based technique because they may prefer to interact with systems that use their own ratings instead of using other users' ratings.

To compute a prediction of an active user for a particular item, the following notations are provided. A set of users is represented as  $U = \{U_1, \dots, U_u\}$ ; a set of items is represented as  $I = \{I_1, \dots, I_i\}$ ;  $R$  is a rating matrix of user-item  $(u, i)$  where  $r_{u,i}$  means rating of a user  $u$  for an item  $i$ ; and a set of possible ratings is represented as  $S$ , where its values take range of numerical ratings  $\{1, 2, 3, 4, 5\}$ . Most CF systems consider the value 1 as "strongly dislike" and the value 5 as "strongly like". It is worth noting that  $r_{u,i}$  should only take one rating value.

After computing the similarity scores between a pair of users or items, two matrices are constructed. The first user-user matrix classifies users who have given similar ratings to a set of items. Similarly,

the second item-item matrix identifies a set of co-rated items by a set of users. The next step is to compute the prediction using (5.1), where  $\hat{r}$  means a predicted rating of a targeted user  $u$  for item  $i$ .

$$\hat{r}(u, i) = \bar{r}_u + \frac{\sum_{v \in s(k)} \text{sim}(u, v) * (r_{v, i} - \bar{r}_v)}{\sum_{v \in s(k)} |\text{sim}(u, v)|} \quad (5.1)$$

where,  $\bar{r}_u$  is the mean rating of user  $u$ ;  $v \in s(k)$  is the number of top  $k$  similar users who have also rated item  $i$ ;  $r(v, i)$  is a rating of the nearest neighbor  $v$  for item  $i$ ;  $\bar{r}_v$  is the mean rating of the nearest neighbor  $v$ ;  $\text{sim}(u, v)$  is the similarity index between a user  $u$  and its nearest neighbor  $v$ . The absolute sign for similarity value is used in the denominator to avoid the negative correlation between a targeted user and  $k$ NNs. If a neighbor  $v$  rates an item above average,  $(r_{v, i} - \bar{r}_v)$  would add to the average rating of the user  $u$ . Moreover, the motivation behind multiplying  $\text{sim}(u, v)$  with  $(r_{v, i} - \bar{r}_v)$  is that if the similarity between a targeted user and its neighbor  $v$  is very high, then a targeted user's rating prediction is heavily affected by the neighbor  $v$ 's ratings and vice versa.

#### *Conventional Similarity Measures*

In recommender systems (RSs), the similarity between a pair of users or items is statistically measured to find out how these pairs are correlated with each other. Researchers introduce several conventional similarity functions such as COS, COR, WCOR, CPC, MSD, JMSD, and Jaccard.

#### Cosine Similarity

COS measures the angle between two rating vectors (users or items) where a smaller angle results in a greater similarity and a smaller similarity is obtained by higher angle. Equation (5.2) is used to compute COS between two rating vectors of user  $u$  and  $v$ .

$$\text{Sim}(u, v)^{\text{COS}} = \frac{\sum_{i \in I(u, v)} R(u, i) \cdot R(v, i)}{\sqrt{\sum_{i \in I(u, v)} R(u, i)^2} \cdot \sqrt{\sum_{i \in I(u, v)} R(v, i)^2}} \quad (5.2)$$

where,  $R(u, i)$  is the stored rating for an item  $i$  given by a user  $u$  and  $I(u, v)$  is a total number of co-rated items of users  $u$  and  $v$ . The value of cosine similarity is 0 to 1, where the value 1 signifies the two vectors are identical and the value zero signifies no correlation between the two vectors. It is noteworthy that COS does not consider some users' tendency to give a lower rating even if they highly like a particular item. Hence, Adjust Cosine (ACOS) function is introduced to address the problem of not considering the user's rating preference, by subtracting the average rating [9].

#### Pearson Correlation Coefficient

COR is used on a set of co-rated items or users. It divides the cross product of overrating or underrating of means by the product of the sum of squares of mean rating difference, as given in (5.3).

$$Sim(u, v)^{COR} = \frac{\sum_{i \in I(u, v)} (R(u, i) - \overline{R(u)}) \cdot (R(v, i) - \overline{R(v)})}{\sqrt{\sum_{i \in I(u, v)} (R(u, i) - \overline{R(u)})^2} \cdot \sqrt{\sum_{i \in I(u, v)} (R(v, i) - \overline{R(v)})^2}} \quad (5.3)$$

where,  $R(u, i)$  is the stored rating for an item  $i$  given by a user  $u$  and  $I(u, v)$  is a total number of co-rated items of users  $u$  and  $v$ . It is noteworthy that values of Pearson correlation coefficient are in the range of (+1 to -1), where +1 means high positive correlation and -1 means high negative correlation.

In the literature, researchers introduce several classical versions of COR to improve the correlation computation. For instance, WCOR, CPC, Sigmoid Function-based on Pearson Correlation Coefficient (SCOR) are used in several RSs [121]. To increase the correlation, CPC allows only pairs of ratings on the same positive or negative side. CPC uses an absolute reference instead of the average rating [119], as given in (5.4).

$$Sim(u, v)^{CPC} = \frac{\sum_{i \in I(u, v)} (R(u, i) - R_m) \cdot (R(v, i) - R_m)}{\sqrt{\sum_{i \in I(u, v)} (R(u, i) - R_m)^2} \cdot \sqrt{\sum_{i \in I(u, v)} (R(v, i) - R_m)^2}} \quad (5.4)$$

where,  $R(u, i)$  and  $R(v, i)$  are the stored rating for an item  $i$  given by users  $u$  and  $v$ .  $I(u, v)$  is a total number of co-rated items of users  $u$  and  $v$ . The median value in a rating scale is denoted by  $R_m$  (e.g., one on the rating scale of ten).

#### Mean Square Difference

MSD is a common measure used in RSs. The estimated ratio of sum square of the difference of ratings on co-rated items divided by the cardinality of ratings in co-rated items is the MSD between two users. Then, the Mean square Similarity is calculated by subtracting MSD from 1, as given in (5.5).

$$Sim(u, v)^{MSD} = 1 - \frac{\sum_{i \in I(u, v)} R(u, i) \cdot R(v, i)^2}{|I(u, v)|} \quad (5.5)$$

where,  $R(u, i)$  and  $R(v, i)$  are the stored rating for an item  $i$  given by users  $u$  and  $v$ .  $I(u, v)$  is a total number of co-rated items of users  $u$  and  $v$ .

#### Jaccard

Jaccard measures similarities between a set of instances. It only considers the total number of items rated by two users instead of the actual ratings, which reveals that the more co-rated items, the more similarity exists. Jaccard is given in (5.6).

$$Sim(u, v)^{Jaccard} = \frac{(I_u \cap I_v)}{(I_u \cup I_v)} \quad (5.6)$$

where,  $I_u$  and  $I_v$  are a set of co-rated items by users  $u$  and  $v$  respectively.

### Jaccard Mean Square Difference

JMSD is introduced to solve the problems of the pervious similarity measures. It combines both MSD and Jaccard, in which Jaccard is used to measure the proportion of the co-rated items and MSD is used to obtain the values of ratings, as given in (7).

$$Sim(u, v)^{JMSD} = (Sim(u, v)^{Jaccard}) \cdot (Sim(u, v)^{MSD}) \quad (5.7)$$

### *Limitations of conventional similarity measures*

Though several similarity measures are introduced by researchers and used by different e-commerce providers for their RSs, there are several limitations identified in various situations. These limitations negatively affect the performance of a RS and lead to inaccurate prediction. Bag et al. [119] comprehensively discuss these limitations.

In the literature, several CF techniques and several conventional similarity models are discussed to classify  $k$ NNs and generate recommendations. Although, the performance of current similarity models still suffers from the data sparsity and limited coverage problems [119]. In the aforementioned problems, the objective of this experimental study is to address the data sparsity and limited coverage problems. Thus, the key contribution is to propose a novel similarity model to identify regions of similarity between users or items in the rating matrix, increase the quality of the recommendation, and improve the prediction accuracy in CF.

### **Proposed Similarity Model**

This section discusses the proposed similarity model. It first discusses the motivation, formally introduces the proposed similarity model, discusses the used rating prediction technique, and finally discusses the time complexity of the proposed model.

### *The Motivation of the Proposed Model*

In the literature, it is claimed that the conventional similarity models used in CF heavily rely on the co-rated items [120]. Therefore, if the similarity scores are obtained by only considering co-rated items, which is called the limited coverage problem, the prediction accuracy and the quality of the recommendation are challenged by this problem. Moreover, the performance of the CF is challenged by data sparsity problem. Consequently, these conventional similarity models fail to consider items rated by a few users. Hence, the proposed similarity model addresses drawbacks and limitations of the conventional similarity models used in CF.

The main motivation of the proposed similarity model is twofold. First, in contrast to the conventional similarity models that only consider users who provide same ratings to a set of items, the proposed

model uses all provided ratings to compute the similarity between a pair of users. Second, the proposed model addresses the limited coverage issue by aligning the ratings to find specific similar regions or patterns in the user-item rating matrix. This approach is known in the literature as a global alignment process for any given sequences [46].

#### *Proposed Rating Alignment-Based Similarity Model*

The CF system used in this chapter to evaluate the proposed similarity model follows four main steps carried out in succession: (1) the most used rating datasets in the field of CF are identified to evaluate the performance of the proposed model; (2) a proposed similarity model is applied to compute the similarity scores between users and to find a list of similar users to a target user; (3) a user-based CF technique is used to predict the ratings of a target user's unrated items is the user-based (a ML technique also is classified by as  $k$ NN algorithm); and (4) the top  $N$  items with the top predicted ratings are recommended to a target user.

To compute the similarity scores between users, the proposed model gives three possible scores when comparing rating vectors for user  $u$  and user  $v$ . The cost scores are matching, mismatching, and gap. These scores are determined when implementing the model using a grid search function. The proposed model then performs the rating alignment after assigning scores to find similar regions or patterns between users in the rating matrix. Hence, the proposed similarity model considers all provided ratings instead of computing the similarity based on co-rated items.

Let the rating vectors for user  $u$  and user  $v$  be  $R_u = [R_1^u, R_2^u, \dots, R_N^u]$  and  $R_v = [R_1^v, R_2^v, \dots, R_N^v]$ , respectively. Each one of these rating vectors belongs to the discrete set  $S$ . Before performing the rating alignment, the cost scores between a pair of rating vectors need to be defined using the similarity score between  $R_i^u$  and  $R_j^v$  (5.8).

$$S(R_i^u, R_j^v) = \begin{cases} M & R_i^u = R_j^v \\ N & R_i^u \neq R_j^v \end{cases} \quad (5.8)$$

where, ( $M > 0$ ) and ( $N < 0$ ) are the cost of match and mismatch of rating, respectively. Then, an alignment matrix  $C$  of size  $N \times N$  is computed where the value of each entry of the matrix is computed by (5.9).

$$C(i, j) = \max \begin{cases} C(i-1, j-1) + S(R_i^u, R_j^v) \\ C(i-1, j) + g \\ C(i, j-1) + g \end{cases} \quad (5.9)$$

where,  $g$  indicates the gap penalty, typically a non-positive value. The rating alignment matrix  $C(N,N)$  contains the similarity scores ( $Sim(u,v)^{RACF}$ ) between two rating vectors. RACF is the short name of the proposed similarity model “Rating Alignment-based Collaborative Filtering”. The computation of the rating alignment matrix  $C(N,N)$  is performed using Dynamic programming algorithm. The computation *algorithm 1* is used to compute  $C(N,N)$  and depicted in Figure 5.1. Given the returned rating alignment matrix that contains the similarity scores for users, the *algorithm 2* is used to compute the prediction for an item  $i$  that a target user  $u$  would rate, as depicted in Figure 5.2. It is noteworthy that  $R$  is the user-item rating matrix.

### Rating Prediction

The next step is to compute the prediction using (5.10), where  $\hat{r}$  means a predicted rating of a targeted user  $u$  for an item  $i$ .

$$\hat{r}(u, i) = \bar{r}_u + \frac{\sum_{v \in s(k)} Sim^{RACF}(u,v) * (r_{v,i} - \bar{r}_v)}{\sum_{v \in s(k)} Sim^{RACF}(u,v)} \quad (5.10)$$

where,  $\bar{r}_u$  is the mean rating of user  $u$ ;  $v \in s(k)$  is the number of top  $k$  similar users who have also rated item  $i$ ;  $r(v,i)$  is a rating of the nearest neighbor  $v$  for item  $i$ ;  $\bar{r}_v$  is the mean rating of the nearest neighbor  $v$ ;  $sim^{RACF}(u,v)$  is the similarity index between a user  $u$  and its nearest neighbor  $v$ .

### Time Complexity

Computing each value of  $C(i,j)$  is a  $O(1)$  operation (which includes two additions, one comparison and one max operations). The complexity of computing an alignment between two users is  $O(N^2)$ . Hence, the time complexity to compute the similarity scores between users in the whole dataset is a  $O(N^2M^2)$ . It is noteworthy that  $M$  and  $N$  are the total number for users and items, respectively.

## Experiments

This section initially discusses the four data sets and the metrics used to evaluate the accuracy of the proposed model. Then, intensive experiments are carried out to validate the superiority of the proposed similarity model based on CF. Moreover, the evaluation process considers comparing the results of the proposed model with other state-of-the-arts similarity functions including CS, COR, MSD, and JMDS. Finally, the setup of the experiments and the results of the experiments are discussed.

---

**Algorithm 1** Algorithm for computation of alignment cost

---

```

procedure COMPUTERACF( $R_u, R_v, g, \mathcal{M}, \mathcal{N}$ )
  for  $k = 0$  to  $N$  do
     $C(k, 0) = g \times k$ 
  end for
  for  $l = 0$  to  $N$  do
     $C(0, l) = g \times l$ 
  end for
  for  $i = 1$  to  $N$  do
    for  $j = 1$  to  $N$  do
       $a = C(i - 1, j - 1) + S(R_i^u, R_j^v)$ 
       $b = C(i - 1, j) + g$ 
       $c = C(i, j - 1) + g$ 
       $C(i, j) = \max(a, b, c)$ 
    end for
  end for
  return  $C(N, N)$ 
end procedure

```

---

Figure 5.1 Algorithm for Alignment Cost Computation

---

**Algorithm 2** Algorithm to predict the rating of  $u$  for item  $i$ 

---

```

procedure COMPUTERATING( $R, u, i, K$ )
  Use user's ratings to build user profile
  for Each user  $v$  in  $R$  do
     $\text{Sim}(u, v) = \text{ComputeRACF}(R_u, R_v, g, \mathcal{M}, \mathcal{N})$ 
  end for
  Find the top  $K$  neighbours ( $S(K)$ )
  Predict the rating of user  $u$  for item  $i$  eq. (10)
  return  $r_{ui}$ 
end procedure

```

---

Figure 5.2 RACF algorithm for rating prediction

*Datasets*

To address the research questions and evaluate the performance of the proposed similarity model, this research identifies the most used datasets from several research papers [39,83]. The researchers discuss the publicly available recommendation datasets. The selection of these particular domains is motivated by [39], since the researchers argue that the most used datasets are from these domains. These datasets

are “Jester,” “MovieLens,” “Book-Crossing” and “Last.fm”. Table 5.1 demonstrates the description and the main characteristics of the used datasets to evaluate the proposed algorithm.

Table 5.1 Description of the used datasets

<b>Data</b>	<b>Jester</b>	<b>MovieLens 1 M</b>	<b>Book-Crossing</b>	<b>Last.fm</b>
<b>No. of ratings</b>	4.1 million	1 million	1.1 million	92,834
<b>No. of users</b>	73,496	6,040	278,858	1,892
<b>No. of items</b>	100	3,592	271,379	17,632
<b>Range</b>	-10 to +10	1,2, ..., 5	1,2, ... , 10	1,2, ..., 5
<b>Domain</b>	Joke	Movie	Book	Music

#### 1- Jester dataset

The Jester is a WWW-based joke RS. The Jester dataset includes 4.1 million ratings entered by 73,496 users for 100 jokes. User ratings are explicitly stored on a real value, ranging from -10 to +10.

#### 2- MovieLens 1 million

The MovieLens dataset is a movie RS. It includes 1.1 million ratings entered by 6,040 users for 3,592 movies. User ratings are explicitly stored on a real value, ranging from 1 to 5.

#### 3- Book-Crossing

The Book-Crossing dataset is a book RS. It includes 1 million ratings entered by 278,858 users for 271,379 books. User ratings are explicitly stored on a real value, ranging from 1 to 5.

#### 4- Last.fm

The Last.fm is a popular music platform. It includes 92,834 ratings entered by 1,892 users for 17,632 items. User ratings are implicitly inferred by composing the number of the listening times into real values of 1 to 5. Equation (5.11) is used to make the required transformation and defined in [83] as follows:



$$r = \begin{cases} \lfloor \log_{10} l \rfloor + 1, & \text{if } \lfloor \log_{10} l \rfloor + 1 \leq 5 \\ 5, & \text{otherwise} \end{cases} \quad (5.11)$$

where  $l$  is the number of listening times,  $r$  is the implicit rating after transforming and  $\lfloor . \rfloor$  is the rounding operator towards zero.

### *Evaluation Metrics*

Accuracy is the most utilized property to examine the performance of different CF algorithms. There are two classes for measuring the accuracy: (1) an accuracy of the ratings predictions, when ratings are available where an algorithm's objective is to learn function  $f: \mathcal{U} \times \mathfrak{I} \rightarrow \mathcal{S}$  that predicts a rating of a user  $u$  for an item  $i$ , and (2) an accuracy of the usage predictions [108], when the objective of CF is to recommend items as the top- $K$  items that it may have a utility for users. It is noteworthy that when researchers evaluate algorithms, they use a different set of instances for evaluation step known as the testing-set, while they use other instances to train algorithms known as the training set. The mean absolute error (MAE), rooted mean squared error (RMSE) are mainly used to evaluate the prediction accuracy and adopted in this empirical study.

MAE is used to estimate the average absolute deviation between the actual ratings and the prediction ratings, MAE is defined in (5.12),

$$MAE = \frac{1}{|\mathfrak{I}|} \sum_{(u,i) \in \mathfrak{I}} |\hat{r}_{ui} - r_{ui}| \quad (5.12)$$

where  $|\mathfrak{I}|$  is the set of recommended items  $i$  from a test set,  $\hat{r}_{ui}$  is a predicted rating for item  $i$  of a target user  $u$ , and  $r_{ui}$  is an actual rating for an item  $i$  of a target user  $u$  from a training set.

RMSE reports the degree of deviation between the predicted ratings and the actual ratings. It heavily penalizes large deviation by squaring the errors before summing them. Lower RSME indicates higher prediction accuracy. RMSE is evaluated in (5.13),

$$RMSE = \sqrt{\frac{1}{|\mathfrak{I}|} \sum_{(u,i) \in \mathfrak{I}} (\hat{r}_{ui} - r_{ui})^2} \quad (5.13)$$

where  $|\mathfrak{I}|$  is the set of recommended items  $i$  from a test set,  $\hat{r}_{ui}$  is a predicted rating for item  $i$  of a target user  $u$ , and  $r_{ui}$  is an actual rating for an item  $i$  of a target user  $u$  from a training set.

### *Experiments Setup*

In order to proceed with the intensive experiments, there are preprocessing steps are carried out: (1) feature selection, (2) normalization using mean centric, and (3) a random sampling using 80/20 rule

where 20% of the data for testing the model and 80% for training the model. Also, the used similarity functions to compute the similarity scores between users are CS, COR, MSD, and UMSD. This study compares the proposed similarity model with the previously mentioned models. Moreover, researchers argue that the performance of RSs is based on the selection of  $k$ NNs [9].

Therefore, this paper considers that  $k \in \{2, 5, 10, 20, 50, 70, 100\}$ . The scores for the alignment parameters (match ( $M$ ), mismatch ( $N$ ), and gap ( $g$ )) are determined by a grid search function. The search range of the different parameters are ( $1 \leq M \leq 5$ ), ( $-3 \leq N \leq 3$ ), and ( $-3 \leq g \leq 3$ ) given the conditions of ( $M > N$  and  $g > N$ ). The best parameters resulting from the optimizer function are found to be  $M=1$ ,  $N=-1$ , and  $g=0$ . The implementation is done in Python, and the library used is a surprise. The computational cost of the proposed similarity model is high; hence, the computation of the Sim ( $v, u$ ) for different pair of users is performed using multi-processing.

### *Results of the Experiments*

Figure 5.3 and Figure 5.4 respectively depict the MAE and RMSE of different similarity functions by considering different values of  $k$  on the movielens dataset. The figures show that the MAE and RMSE improve while increasing the value of  $k$ . It is noteworthy that the RACF is less sensitive to the value of  $k$ . Also, the figures indicate that the proposed model RACF performs better than other similarity metrics considering the different values of  $k$ . It is noteworthy that when  $k > 5$ ; the MAE for COR performs poor compared to all other functions. In case of  $k > 10$ , the performance of JMSD and CS is about the same. In case  $k > 20$ , the RMSE of CS, COR, MSD, and JMSD does not change.

Figure 5.5 and Figure 5.6 respectively depict the MAE and RMSE of different similarity functions by considering different values of  $k$  on the jester dataset. The MAE and RMSE show that the performance of COR is poor compared to other similarity functions. The results of the MAE reveal that CS outperforms other functions when  $k=5$ ; the performance of RACF is about the average due to the fact MAE is not sensitive to the outlier; and MSD does perform better than other similarity metrics while  $k \geq 10$ . On the other hand, RACF performs better than other similarity metrics when measuring the accuracy using RMSE. Moreover, RMSE shows that the performance of CS and JMSD is exactly the same when  $k \geq 20$ .

Figure 5.7 and Figure 5.8 respectively depict the MAE and RMSE of different similarity functions by considering different values of  $k$  on the Last.fm dataset. The MAE and RMSE show that the prediction error of CS, COR, MSD, and JMSD significantly drops while  $k > 2$ , and does not improve while  $k \geq 20$ . Moreover, both MAE and RMSE show that MSD outperforms other similarity functions considering the different values of  $k$ .

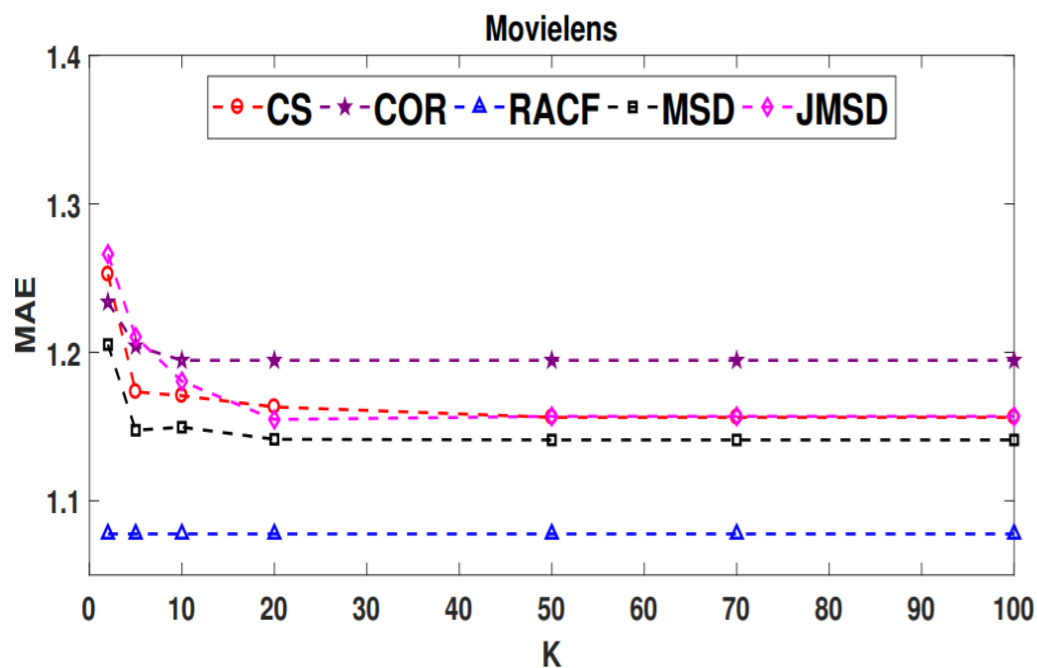


Figure 5.3 Performance comparison of RACF and other similarity functions on MAE@K for Movielens dataset

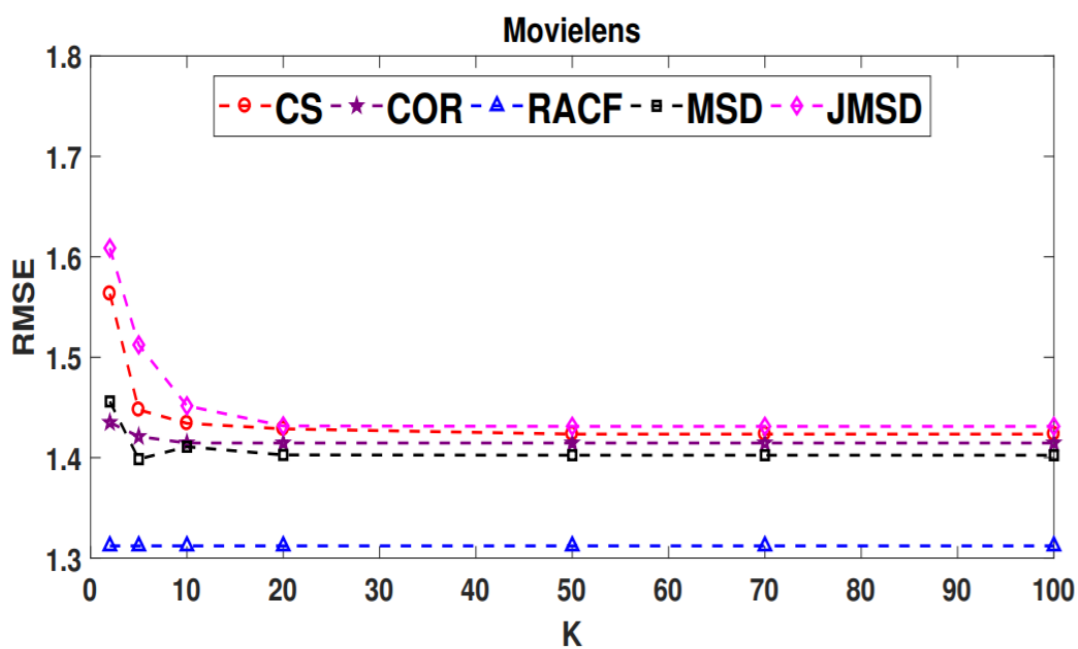


Figure 5.4 Performance comparison of RACF and other similarity functions on RMSE@K for Movielens dataset

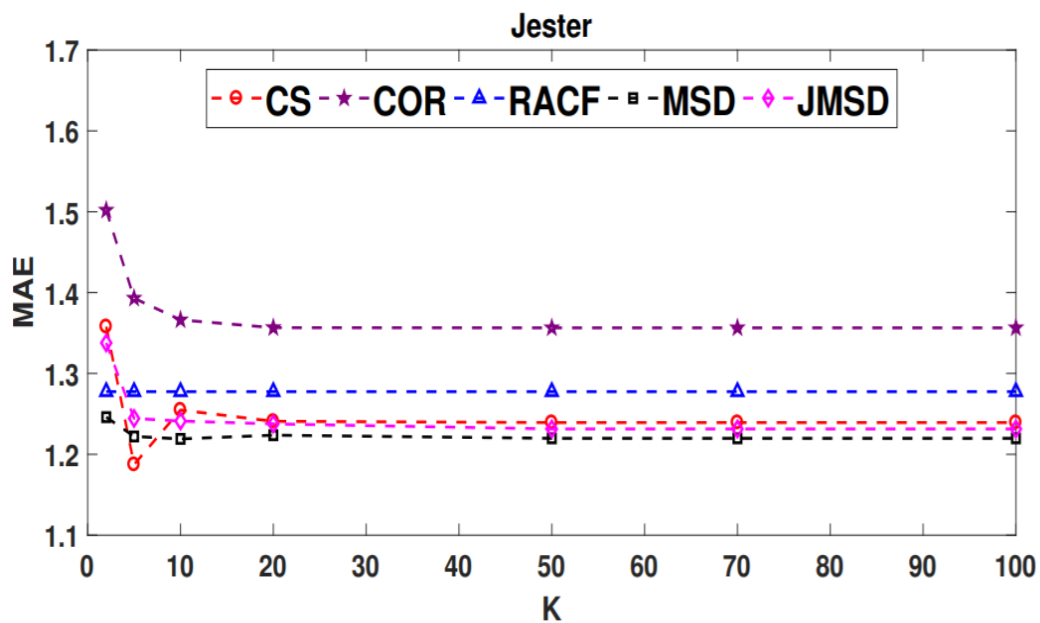


Figure 5.5 Performance comparison of RACF and other similarity functions on MAE@ $K$  for Jester dataset

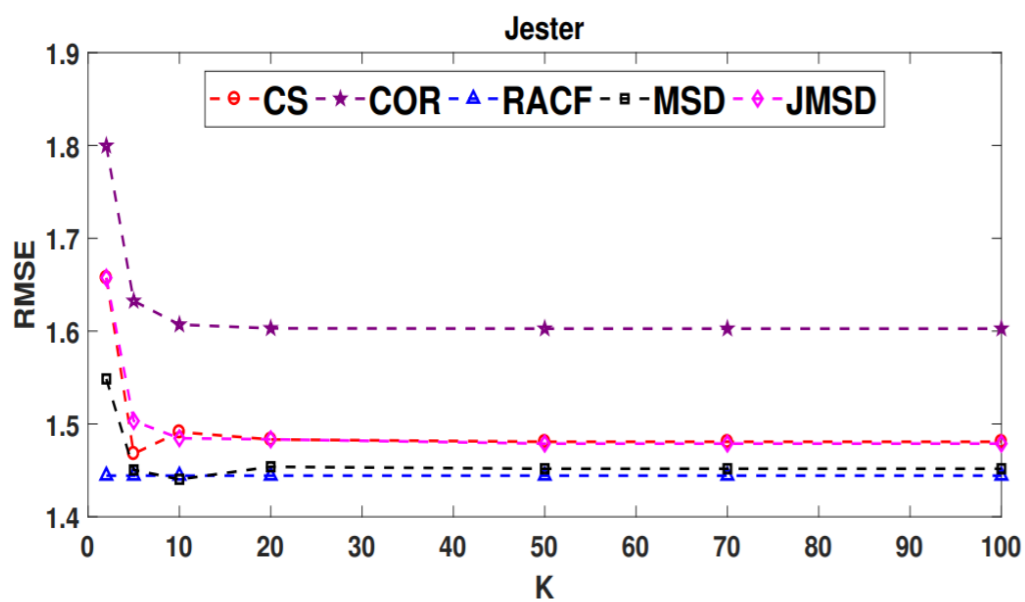


Figure 5.6 Performance comparison of RACF and other similarity functions on RMSE@ $K$  for Jester dataset

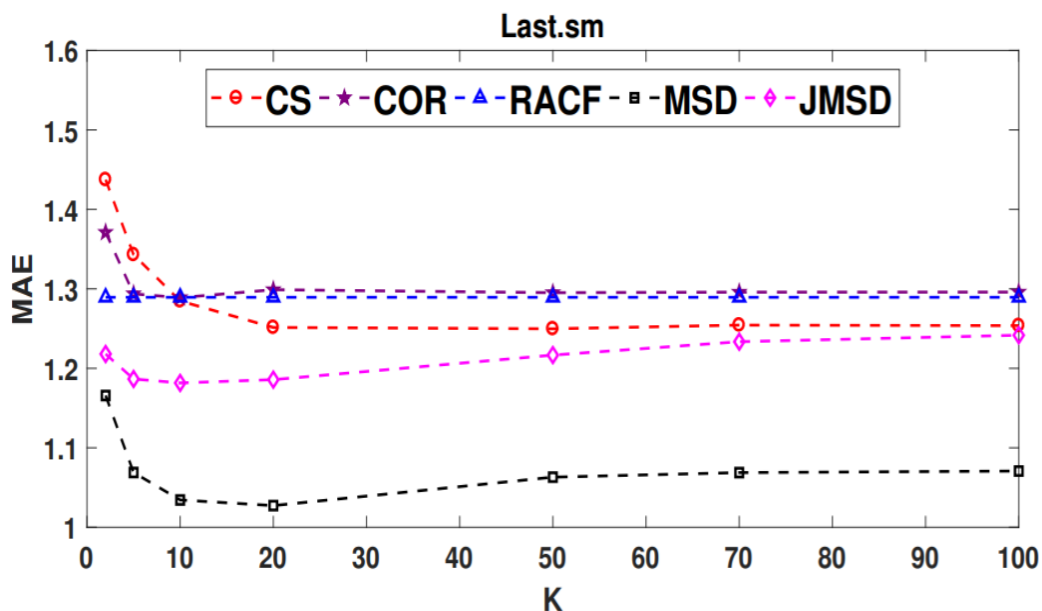


Figure 5.7 Performance comparison of RACF and other similarity functions on MAE@K for Last.Fm dataset

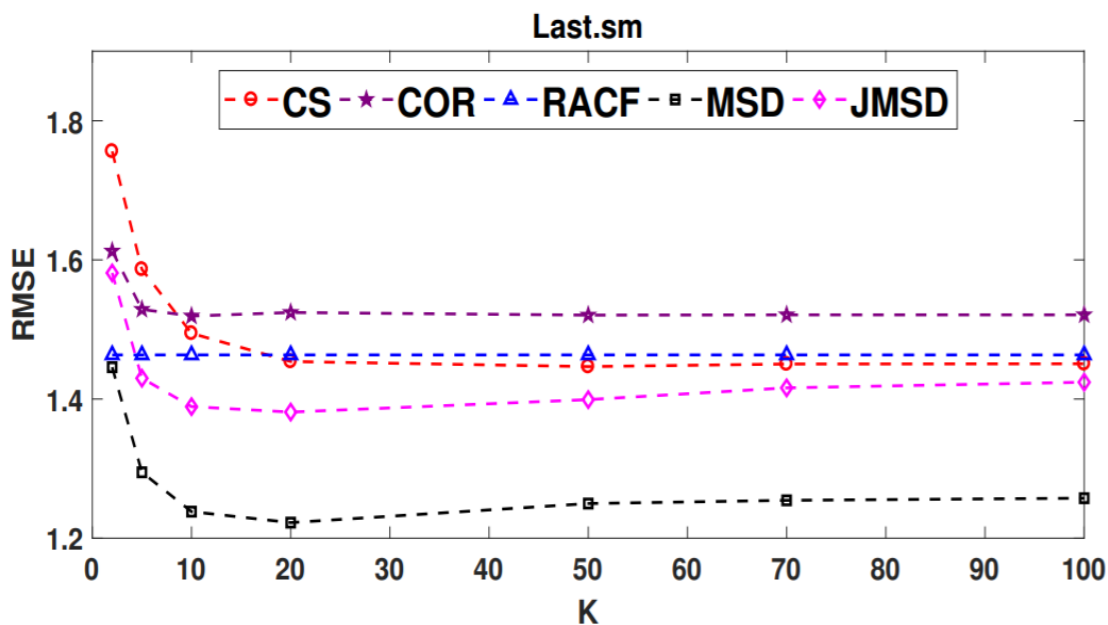


Figure 5.8 Performance comparison of RACF and other similarity functions on RMSE@K for Last.Fm dataset

Figure 5.9 and Figure 5.10 respectively depict the MAE and RMSE of different similarity functions by considering different values of  $k$  on the book-crossing dataset. RMSE reveals that RACF outperforms other similarity functions for all values of  $k$ . Both MAE and RMSE show the poor performance of

JMSD compared to other functions. Figure 5.9 shows that CS performs better than other similarity functions when  $k=2$ . While there is a big gap in the error between RACF and MSD when using RMSE, the MAE shows that RACF and MSD perform about the same, and the performance of MSD does not improve when the value of  $k$  increases.

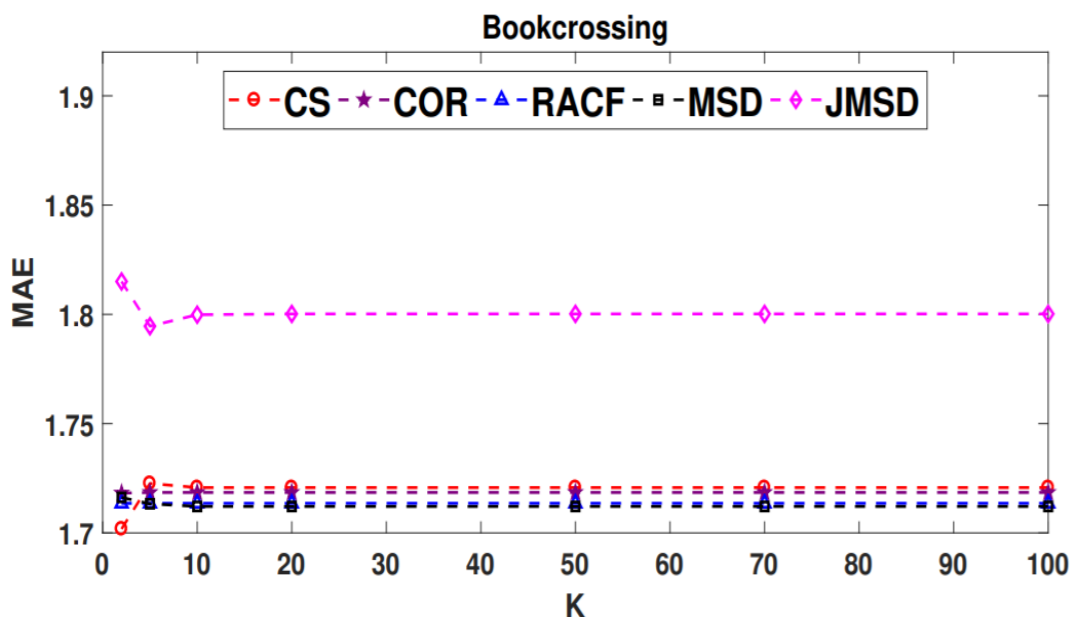


Figure 5.9 Performance comparison of RACF and other similarity functions on MAE@ $K$  for Book-Crossing dataset

Yet, RMSE shows the superiority of RACF compared to CS, COR, and JMSD when  $k=2$ , while the performance of RACF is better than CS and COR when considering MAE. Even though RACF is not able to outperform MSD while using last.fm dataset, the main advantage is that RACF can effectively generate predictions using a minimum number of  $k$ NNs for some cases when there are no similar users to a target user. It is noteworthy that MSD outperforms all other similarity functions because last.fm is not sparse compared to the other used datasets. The ratings in the last.fm are implicitly generated by composing the number of the listening times into real values of 1 to 5.

Table 5.2 discusses the comparison of the MAE and RMSE for the different similarity functions averaged across all values of  $k$ . The reported results from the table using MAE and RMSE indicate that on average, the RACF performs better in movielens and book-crossing datasets. In the case of jester dataset, on average RACF achieves better RMSE, and the MSD achieves better MAE. It is also claimed that on average MSD outperforms all other similarity functions, while RACF on average performs better than CS and COR.

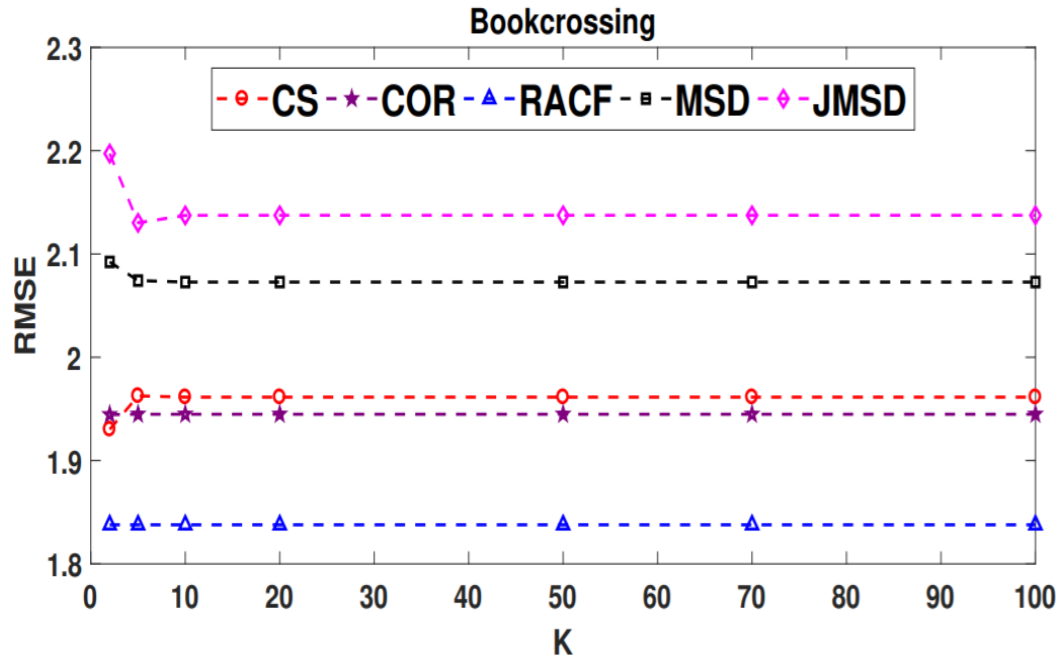


Figure 5.10 Performance comparison of RACF and other similarity functions on RMSE@K for Book-Crossing dataset

Although researchers claim that the optimal number of  $k$  can range from (20 to 50) [25], the main advantage of RACF is that it is not sensitive to the number of  $k$ NNs. RACF is claimed to be an appropriate similarity function when there are few users similar to a new user who does not have enough ratings.

Moreover, the results of [120] show that MSD performs poorly compared to other seven similarity functions in Movielens dataset when  $k \leq 60$ , while MSD in this research shows its ability to decrease the error in the prediction using the same dataset compared to CS, COR, and JMSD similarity functions.

Because it is difficult to assess the effectiveness of the proposed similarity model and to better quantitatively evaluate the overall performance of the different similarity functions compared to the RACF, this study considers another type of evaluation. It compares the MAE and RMSE when considering  $k=2$  as a hand-picked anecdotal case. Therefore, this study claims that on average RACF outperforms other used similarity functions using a set of different datasets collected from popular domains.

Table 5.2 Average performance comparison of RACF and other similarity functions on MAE and RMSE for four datasets

Dataset	Evaluation Metric	CS	COR	RACF	MSD	JMSD
Moveilens	MAE	1.45	1.42	<b>1.31</b>	1.41	1.47
	RMSE	1.18	1.20	<b>1.08</b>	1.15	1.18
Jester	MAE	1.51	1.64	<b>1.44</b>	1.46	1.51
	RMSE	1.25	1.38	1.28	<b>1.22</b>	1.25
Last.fm	MAE	1.52	1.54	1.46	<b>1.28</b>	1.43
	RMSE	1.30	1.31	1.29	<b>1.07</b>	1.21
Book-crossing	MAE	1.96	1.94	<b>1.84</b>	2.08	2.14
	RMSE	1.72	1.72	<b>1.71</b>	<b>1.71</b>	1.80

Figure 5.11 shows the MAE comparison when  $k = 2$  for different similarity functions using different datasets. In the case of the movielens dataset, the performance of RACF is better than all the other functions, while MSD outperforms other functions on both jester and last.sm datasets. It is noteworthy that the performance of CS, COR, RACF, and MSD is about the same on the book-crossing dataset. On average across four datasets, however, the performance of MSD and RACF is about the same.

Figure 5.12 shows the comparison RMSE when  $k = 2$  for different similarity functions using different datasets. The figure indicates that the performance of RACF is better than all the other similarity functions in movielens, jester and book datasets. In case of last.sm, however, MSD and RACF perform about the same. On average across four datasets, the performance of RACF is better than all other similarity functions.

### Discussion

This empirical study proposes a novel rating alignment-based similarity function used for CF. This model addresses the problems of data sparsity and co-rated items (a.k.a. limited coverage problem) in CF. Therefore, using the proposed similarity model in CF can effectively enhance the performance of the accuracy and improve the quality of the recommendation. The results of this study show that RACF can perform better compared to other similarity functions for sparse datasets such as movielens.



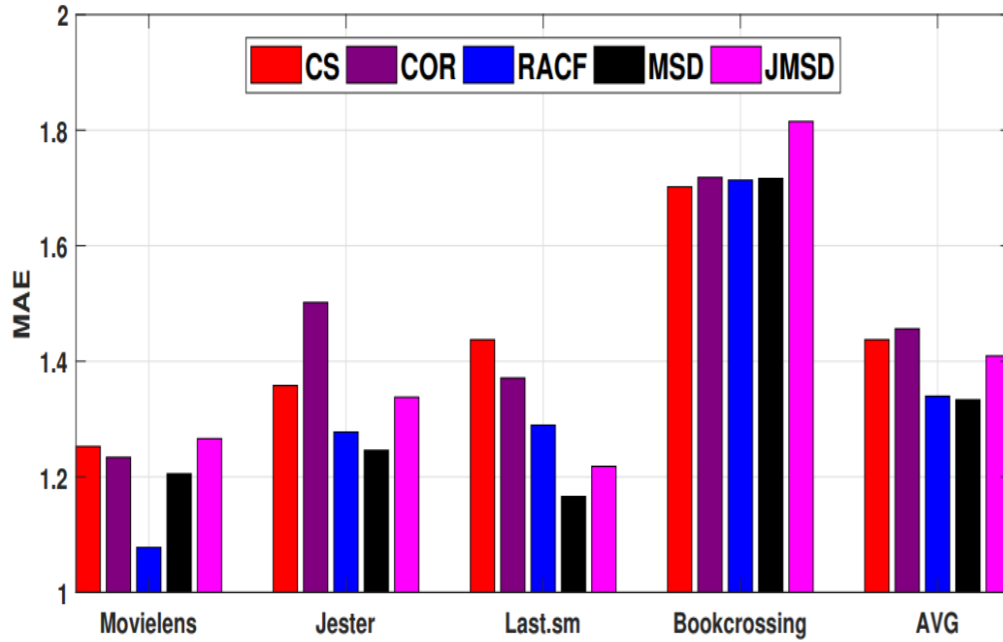


Figure 5.11 Performance comparison of RACF and other similarity functions on MAE@2 for the different datasets

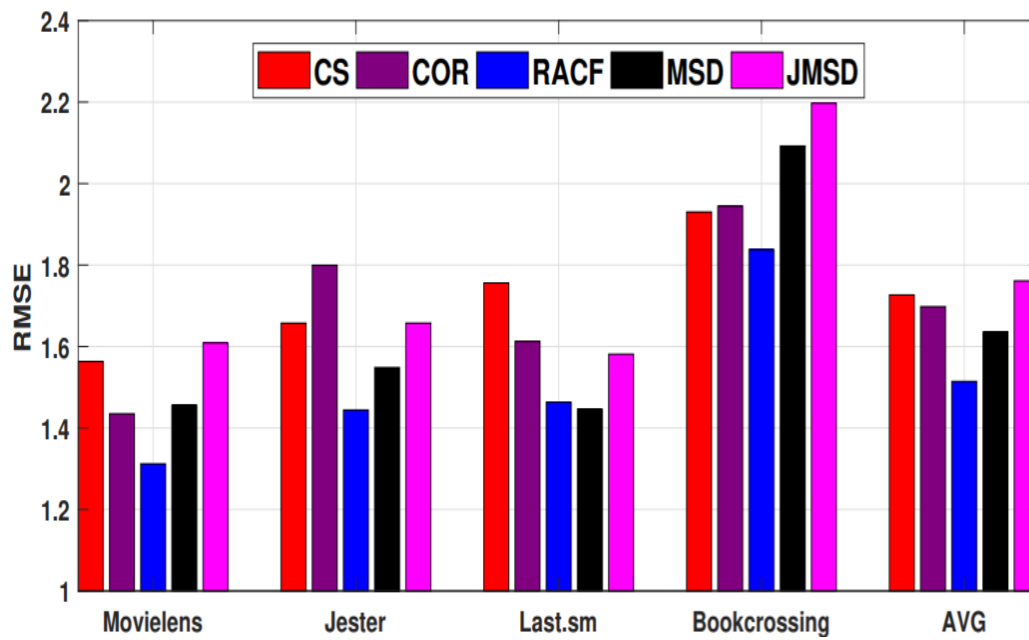


Figure 5.12 Performance comparison of RACF and other similarity functions on RMSE@2 for the different datasets

Moreover, this study claims that on average RACF achieves a significant improvement in the accuracy using RMSE compared to other similarity functions. This indicates that RACF is performing well on the outlier users. Those outlier users might have fewer co-rated items and the conventional similarity functions are failing to estimate the accurate ratings, but the proposed RACF can still perform well on those users. It is worth noting that the proposed similarity model depends on the extract ordering on the rating vector, while conventional similarity functions do not depend on the ordering of items in the user-item matrix. Therefore, to optimize use of the proposed similarity model, it is recommended to sort the items in a meaningful manner depending on the dataset. For instance, the items in a movie RS can be sorted according to the genre of a movie, or the items in a book RS can be sorted based on the type of a book.

The proposed similarity function enhances the performance of CF by improving the accuracy compared with all other used similarity functions. The RMSE and MAE of the proposed similarity model reduce 7.52% to 12.12% and 6.93% to 11.5% respectively on Movie-lens dataset; the RMSE of the proposed model reduces at least 1.37% to 13.35% on Jester dataset; and the RMSE of the proposed model reduces at least 16.71% to 5.82% on Book-crossing dataset.

## **Conclusion**

The current research gaps in the field of RSs include methods of addressing data sparsity and limited coverage. These problems can negatively affect the performance of CF. Moreover, developing a CF system while not paying close attention to the similarity function used to compute the similarity between users is an insignificant methodology in the domain of RSs. Still, there are intensive attempts by researchers to improve the performance of CF in terms of similarity computation. Thus, this empirical study proposes a rating alignment-based similarity model while considering all ratings values in a rating vector. In contrast to other similarity functions, the proposed model does not depend on co-rated items to compute the similarity between users. The proposed similarity model instead takes into account considering un-co-rated and un-rated items by assigning scores for matching, mismatching, and gap in the rating vectors of users. To experimentally validate the effectiveness of the proposed similarity model, this study uses four popular datasets of Movielens, Jester, Last.fm, and movielens. Results obtained from MAE and RMSE indicate that the proposed similarity function can successfully improve the quality of recommendation and the accuracy of the prediction when the data is sparse.

This study generally contributes to the domain of RSs and in particular to the field of CF. The main contribution is twofold. First, the proposed similarity model can be used to address the main drawbacks of CF. These drawbacks are data sparsity and limited coverage. Moreover, the proposed model makes

use of all the stored ratings to accurately compute the similarity between users. Second, this study experimentally examines the effectiveness of the proposed model by comparing its performance with state of the art similarity functions CS, COR, MSD, and JMSD. On average, it is argued that the proposed similarity model shows superior performance to other previously discussed similarity functions.

The main advantage of the proposed similarity model is that it is less sensitive to the total number of similar users or nearest neighbors to a target user. It can generate recommendation and make prediction while considering  $k=2$  in the worst case, while most of the conventional similarity measures heavily depend on a higher number of  $k$ , which increases the time complexity of these models.

## Chapter 6: A Personalized Academic Advisory Recommender System

*This chapter is based on:*

**A. Althbiti**, S. Algarni, T. Alghamdi, and X. Ma, “A Personalized Academic Advisory Recommender System: A Case Study,”. *Proceedings of 2021 The Asia Pacific Computer Systems Conference*, In Springer.

### Introduction

Recommender systems (RSs) are novel statistical techniques developed to retrieve useful information and to generate predictions based on provided data from users [1]. These systems can reduce the information overload and the number of the retrieved selections. They systematically personalize the recommendation process based on each user’s needs. RSs are used in several domains including e-government, e-learning, e-commerce, e-business, e-library, and e-resources [36].

RSs have been also used to personalize the education process, in particular to recommend a list of courses to students. Since students strive to find a list of courses that match their interests and learning objectives, they seek recommendation from their advisors. Hence, the traditional system of recommending courses to students is time consuming, risky, and monotonous work. These drawbacks may negatively affects both student performance and the learning experience. In addition to this challenge, in some cases students may face an academic hold when registering for courses due to a lack of communication with their advisors.

Using RSs to automate the process of courses recommendation is crucial to help students find a list of courses that support their learning objectives and also to help advisors or professors to offer courses that students need. Moreover, this system can reduce the effort and the time that advisors spend with each student to find a list of courses each semester. By only collecting primary learning objectives from each student, it is possible to build a recommender system that automates the process of course recommendation. In particular, the content-based recommender system is an effective approach to build this model. This approach relies on item and user descriptions (content) to build item representations and user profiles to recommend items similar to an active user’s previously preferred items [20].

To apply content-based filtering for course recommendation, a list of courses and their descriptions are collected to build courses’ description space. These descriptions are transformed from unstructured to structured representations. With the development of machine learning (ML) algorithms, current content-

based RS learns profiles for students [118]. Moreover, using a ML technique to learn these profiles is the main component of content-based filtering. Hence, the introduced system can compare a student's profile and a course's description and then recommend a list of courses for an active student.

The structure of this chapter is as follows. It first discusses the related research works of this research. Then, it discusses the introduced system and the proposed approach. Then, it discusses the implementation of the framework of the proposed system. Then, it discusses the experiment and the evaluation. Finally, it discusses the conclusion.

### **Literature Survey**

Providing a personalized list of items ultimately reduces the number of the retrieved items. Subsequently, users are exposed to a list of items that share features of their previously preferred items. Hence, it is rational to develop RSs that support educational objectives for students and instructors. This section provides a comprehensive review on educational recommender systems with the focus on course recommender systems.

Researchers in [133] proposed a model that aims to find the best combination of courses in E-Learning. They used four association rule algorithms: Apriori Association Rule, PredictiveApriori Association Rule, Tertius Association Rule and Filtered Associator. The main drawback of their model is that they did not consider students' profile to personalize the learning process. Another study was conducted to examine the impact of encouraging user contribution in the context of CourseAgent, a community-based course recommender system [122]. The recommendation power of CourseAgent is based on course ratings submitted by a group of students. To increase the number of course ratings, CourseAgent uses an incentive approach which turns user feedback into a self-beneficial activity.

Researchers claimed that conventional e-Learning environments are based on static contents considering that all students are similar, so these traditional systems are not able to respond to each student's needs [123]. They also pointed out that new educational systems should be introduced to certify the personalization of learning contents. In their work, the aim is to develop a new personalization approach that exposes to students the best learning resources according to their preferences, background knowledge, and their memory capacity to store information. They introduced a new recommendation method based on collaborative and content-based filtering: NPR\_eL (New multi-Personalized Recommender for e Learning). They integrated this approach in a learning environment to deliver personalized learning resources. Finally, they validated the effectiveness of their

approach through the design, implementation, analysis and evaluation of a personal learning environment.

Another study was conducted to introduce a novel recommendation architecture that is able to recommend interesting post messages to the students in an e-learning online discussion environment based on semantic content-based filtering and learners' negative ratings [124]. The proposed system was evaluated against existing e-learning recommender systems that use similar filtering techniques in terms of recommendation accuracy and students' performance. The results demonstrated that the learning performance has been improved for the students supported by recommendations based on their proposed technique as compared to other similar recommendation techniques.

RSs and big data analytics techniques are being used to analyze the massive educational data and generate different predictions and recommendations for students, instructors and universities. Researchers used RSs for big data in education [125]. In particular, they use collaborative filtering (CF) based recommendation techniques to recommend elective courses to students, depending upon their grade points gained in other subjects. They used item based recommendation of Mahout ML library on top of Hadoop to get a set of recommendations. Similarity Log-likelihood was used to find similar patterns among students' grades and subjects. The conclusion of their study argues that the recommendations generated by educational RSs can be useful to the educational sectors to improve the performance of students and instructors.

Some RSs are designed to provide advisory guidance regarding the selection of courses. For instance, PCRS, a short name for Personalized Course Recommender System Based on Hybrid Approach, uses ontologies that retrieve a list of useful courses along with content-based RSs to generate accurate recommendation [126]. The researchers developed a hybrid RS that can be integrated to improve the effectiveness of any E-learning system, to ease information access and to provide personalization to students. The results of the experiments in their research revealed that using RS to select courses performs well compared to the conventional methods.

Several RSs are introduced to provide educational guidance regarding the management of learning objects such as courses and exercises. For instance, a smart course recommender system was proposed in [129] to provide instructors with recommendations to help them better manage their courses based on the various learning styles of students. Moreover, a personalized group-based recommendation system was developed to improve students' search experience on the Web based on their behavior patterns and skills [130]. The researchers developed an adaptive RS that comprises dynamic profiling

and content re-ranking mechanisms that cater to students in finding Web-based learning resources based on their academic records and learning activities.

Other research presents a Personalized Career-path Recommender System (PCRS) to offer guidance and assist high school students choose engineering discipline [127]. Their system can overcome the problem when high-school students decide to choose a university specialization. The design of PCRS uses fuzzy intelligence of N-layered architecture and uses students' academic performance, personality type, and extra-curricular skills. The results indicated that a slight agreement between the suggested recommendations of PCRS and the actual discipline of the research sample.

This literature review of related research in personalized course RSs provides more understanding of the different aspects posed by RSs researchers to enhance the learning experience of students and instructors. For a clearer view of the different types of personalized RSs, Table 6.1 summarizes the different RSs and their implementation approaches, platform, evaluation, dataset and results. The literature review demonstrates that using RSs is very significant nowadays because they efficiently help students discover interesting courses from a vast amount of available options. However, most of these studies do not use students' profiles and courses' descriptions to make better recommendations.

Hence, a Personalized Academic Advisory Recommender System (PAARS) is introduced which uses a content-based filtering and an ensemble learning algorithm of k-nearest neighbors (*k*NN) classifier and Vector Space Model (VSM) to generate a list of recommendations. For additional information about educational recommenders, see [128, 131, 132].

### **Approach**

Students often prefer to take courses that offer knowledge of a specific topic and prepare them for the career development. They often seek guidance from their advisors to find courses from different disciplines that they need to take before graduation. However, it happens sometimes that a student only enrolls in several courses that have no correlation with his/her interests. Yet, if a student has a specific goal or profile, it is possible to personalize and automate the process of course recommendation. There are two fundamental mechanisms used to build this system: (1) a content-based technique and (2) an ensemble learning algorithm of data mining technique and VSM.

In particular, a *K*NN classifier, a supervised ML algorithm that stores training records in memory and classifies a new unseen item by comparing it to all stored items by using a similarity function , is

Table 6.1 Summary of different Learning-based RSs and their implementation approaches, platform, evaluation, dataset and results

Title	Purpose	Approach	Platform	Evaluation	Dataset	Result
A Comparative Study of Association Rule Algorithms for Course Recommender System in E-learning [133]	Recommending the best combination of courses	Association rule algorithms: Apriori Association Rule, Predictive Apriori Association Rule, Tertius Association Rule & Filtered Associator	N/A	Experiment Using WEKA	The real data from Moodle course of Walchand Institute of Technology at Solapur University India	Apriori association algorithms perform better than the others
Personalized recommender system for e-Learning environment [123]	Recommending learning materials	Based on CF and content-based filtering	Software integrated in a learning environment	Experiment	Book-Crossing and their University's dataset	Producing personalized recommendations of a good quality
Utilizing Learners' Negative Ratings in Semantic Content-based Recommender System for e-Learning Forum [124]	Introducing a novel recommendation architecture	Based on semantic content-based filtering and learners' negative ratings	Web based	Experiment	N/A	Their proposed e-learning recommender system outperforms other similar e-learning recommender systems



Recommender system for big DATA IN EDUCATION [125]	Recommending academic choices	CF based recommendation techniques	N/A	Mahout machine learning library on top of Hadoop	Students' grades	Identifying applicability of recommender system for huge size of educational data
PCRS: Personalized Course Recommender System Based on Hybrid Approach [126]	Recommending academic choices	Hybrid approach with ontology	Web based	Experiment	N/A	The proposed recommender systems would perform better compared to other educational personalized RSs
A Personalized Group-Based Recommendation Approach FOR WEB SEARCH IN E-LEARNING [130]	Recommending learning materials	Based on content filtering and decision tree C <sub>4.5</sub> algorithm	Web based	Experiment	N/A	The proposed system augments the Google search engine

used to learn a student's profile [20]. The use of *KNN* classifier is motivated by [20], as the researchers claim that it performs better than other statistical learning methods where the computed probability is not important. Moreover, another advantage of the *kNN* approach is that it is very efficient and easy to implement compared to other learning methods.

### *Content-Based Filtering*

Content-based filtering is a widely used approach in RSs. It aims to recommend a list of items similar to those a given student has liked in the past. Indeed, the basic process performed by a content-based filtering consists in matching up the features of a student's profile in which preferences are stored with the features of a course in order to recommend to a student new interesting courses, as is the case in this research. Using a content-based RS allows us to analyze a set of courses and its descriptions, to calculate the similarity between courses, and to build a profile of student interests based on the features of the preferences collected from that student [20]. The learned profile is a structured representation of a student's interests used to expose a student to new interesting courses. As mentioned previously, several techniques are considered to represent the courses, to model a student's profile, and to compare a student's profile with a course's representation. The framework architecture of the integrated model in PAARS is depicted in Figure 6.1. The fundamental process of the developed model is carried out in three basic levels handled by a separate module:

#### (1) Description Analyzer

Description analyzer module is used to transform description of courses from unstructured (text) form to structured form by extracting features *n*-grams. It is considered as a pre-processing step to extract structured relevant information from descriptions of courses [20]. It represents the descriptions of the courses in a form suitable for the following processing steps. The description of courses is analyzed by feature extraction techniques that usually use techniques from Information Retrieval (IR) systems to map the course description from the original *Description Source* space to the target one. Hence, the unstructured collected description of each course *C* is mapped to construct a vector  $V_{C_i}$  and is stored in the repository *Represented Courses*. This representation is the input to the PROFILE LEARNER and FILTERING COMPONENT.

Since courses that can be recommended to students are represented by a set of features (name, department, and description), there are many benefits of representing courses' description as VSM. For instance, Cosine Vector similarity can be used and even be extended to use *KNN* classifiers. VSM with

basic term frequency-inverse document frequency (TF-IDF) weighting is a spatial representation of text documents used for weighting the terms and measuring the feature vector similarity [20].

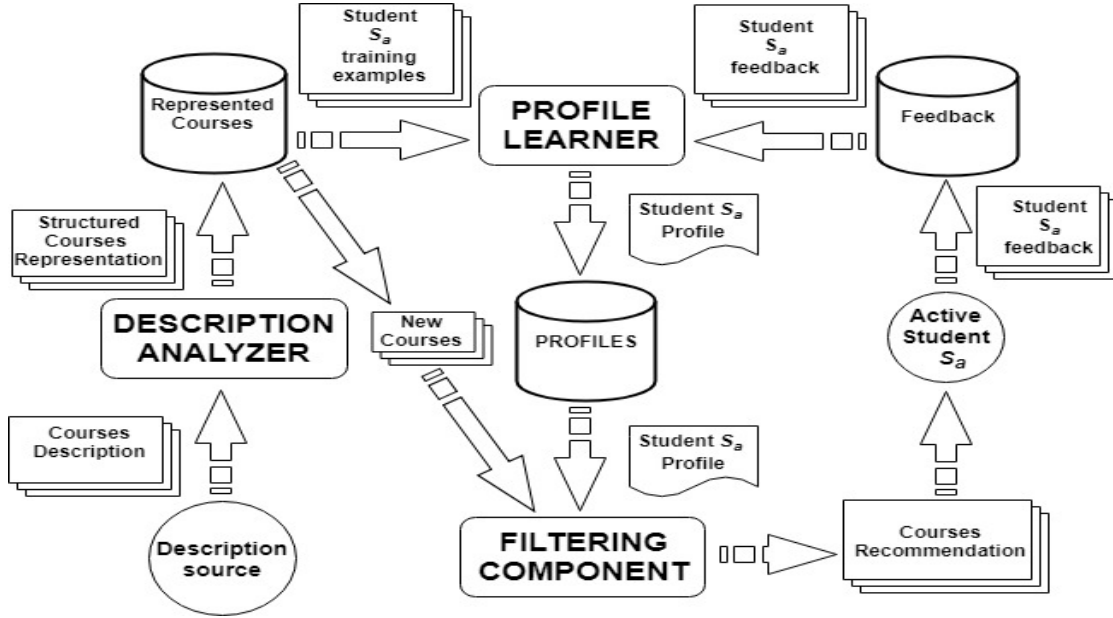


Figure 6.1 Framework architecture of the proposed model

TF represents the frequency a word appears in the description of a course. The main idea of IDF is that the term will be less important if appears more in description of other courses. Formally, every course description is represented as a vector of term weights, where each weight specifies the degree of association between the description and the term. Let  $D = \{d_1, d_2, \dots, d_N\}$  denote a set of descriptions, and  $T = \{t_1, t_2, \dots, t_n\}$  be the set of words in a description.  $T$  is obtained by applying some standard natural language processing (NLP) operations, such as punctuation and capitalization, lemmatization, tokenization, stop-words removal, and stemming [134]. Each description  $d_j$  is represented as a vector in a  $n$ -dimensional vector space, so  $d_j = \{w_{1j}, w_{2j}, \dots, w_{nj}\}$ , where  $w_{kj}$  is the weight for term  $t_k$  in description  $d_j$ . Equations (6.1), (6.2), and (6.3) are introduced to compute TF-IDF.

$$TF - IDF(t_k, d_j) = TF(t_k, d_j) \cdot \log \frac{N}{n_k} \quad (6.1)$$

where  $N$  signifies the total number of descriptions, and  $n_k$  signifies the number of descriptions in which the term  $t_k$  appears at least once.

$$TF(t_k, d_j) = \frac{f_{k,j}}{\max_z f_{z,j}} \quad (6.2)$$

where the maximum is computed over the frequencies  $f_{z,j}$  of all terms  $t_z$  that appear in description  $d_j$ .

$$IDF = \log \frac{N}{nk} \quad (6.3)$$

In order for the weights to fall in the range of [0, 1] interval and for the description to be represented by vectors of equal length, weights gained by (1) are often normalized by cosine normalization (6.4) which enforces the normalization assumption,

$$w_{k,j} = \frac{TF-IDF(t_k,d_j)}{\sqrt{\sum_{s=1}^{|T|} TF-IDF(t_s,d_j)^2}} \quad (6.4)$$

As was specified previously, a similarity measure is essential to determine the affinity between two descriptions. Several similarity measures have been used to describe the closeness of two vectors; among those measures, cosine similarity (6.5) is the most widely used [20],

$$sim(d_i, d_j) = \frac{\sum_k w_{ki} \cdot w_{kj}}{\sqrt{\sum_k w_{ki}^2} \cdot \sqrt{\sum_k w_{kj}^2}} \quad (6.5)$$

## (2) Profile Learner

Profile learner module is used to collect data representative of a student's preferences and aims to generalize this data in order to learn a student's profile. Usually, the generalization strategy is learnt through  $k$ NN, which is able to learn a model of student preferences starting from course taken or not taken in the past. To learn and update a profile for an active student  $S_a$ , a student's feedback to recommended courses is collected through the PAARS website and stored in the repository *Feedback*.

This feedback and the related courses descriptions are used during the process of learning a model for a student profile. This model is useful for predicting the actual relevance of newly recommended courses. Students can also explicitly define their areas of interests as an initial profile without providing any feedback. Hence, two different techniques are adopted to explicitly record students' feedback. The first one is when a student uses PAARS website and provides his/her areas of interests. The second one is when PAARS website requires a student to explicitly evaluate courses. These techniques have the advantage of simplicity and can help students to express their satisfaction or dissatisfaction regarding recommended courses.

To build a profile for an active student  $S_a$ , the training set  $TR_a$  for  $S_a$  is defined.  $TR_a$  is a set of pairs  $(C_n, R_n)$ , where  $R_n$  is the rating provided by a student  $S_a$  on a course representation  $C_n$ . Given a set of course representation labeled with ratings, the PROFILE LEARNER uses  $k$ NN (supervised learning algorithm) to learn a predictive model (*student  $S_a$  profile*) which is usually stored in a *Profile Repository* for later use by the COURSE FILTERING.

### (3) Course Filtering

Course filtering module performs two main functions. It computes the similarity between an active student's  $S_a$  profile and a list of courses to be recommended and filters out courses used to compute the recommendation of a new course for an active student. It is noteworthy that a profile of an active student is also defined as a vector  $V_{S_a}$ . The result is a binary relevance judgment computed using cosine vector similarity measure. The similarity is computed between students' profiles vector  $V_{S_a}$  and a list of courses' profile vectors  $V_{C_n}$ .

Given a new course representation, the COURSE FILTERING predicts whether it is likely to be of interest for an active student  $S_a$ , by matching features in a course representation to those in the representation of a student's preferences (stored in the student profile) and by combining their weights. It also uses a technique to rank potentially interesting courses according to the relevance with respect to a student profile. The ranking process is performed through combining these weights and then gets the final score. Hence, the system recommends the top courses in  $L_c$  to students.

Top-ranked courses are presented in *Courses Recommendation  $L_c$* , that is presented to an active student  $S_a$ . Students' preference often change over time; hence up-to-date information are provided to the PROFILE LEARNER. More feedbacks (ratings) are collected on generated recommendations by letting students express their satisfaction or dissatisfaction with recommendations in  $L_c$ . Afterward, the learning process is implemented again on the new training set, and the resulting profile is used to a student's updated preferences. This process allows PAARS to take into account the dynamic nature of students' preferences.

#### *k*-Nearest Neighbors (*k*NN)

$k$ NN is a supervised machine learning used to build a model for classification problems [118]. It is used in the task of inducing content-based profiles and is well-suited for text categorization [20]. While using  $k$ NN, an inductive process automatically learns a text classifier by learning the features of the categories from a set of *training documents* (in our case training descriptions). The problem with learning a student

profile is considered a binary text categorization task. Each course has to be classified as interesting or not with respect to students' preferences. Moreover, creating a model of a student's preference from the previous collected data is a form of classification learning. Hence,  $k$ NN is used to learn a function that models each student's interests. Given a new course and a student model (student  $S_a$  profile), the function predicts whether the student would be interested in this course.

$k$ NN simply stores training data in memory and classifies a new unseen course by comparing it to all stored courses by using a cosine vector similarity function, since courses are represented using the VSM. If the nearest neighbor courses are determined, the class label for the unclassified course is determined from the class labels of the nearest neighbors. It is noteworthy that the range of the similarity is between -1 and 1, where -1 means that the direction of the two vectors are totally opposite and 1 means these vectors are in the same direction. When the value of the cosine similarity is 0, it means that these two vectors have no relationship. For text matching, the weights are non-negative; hence, the range should be 0 to 1 in our case. The selection of  $k$ NN algorithm is motivated by [135].

### **PAARS Implementation**

The overall system architecture for the PAARS is depicted in Figure 6.2. The framework architecture is summarized into four components:

- (1) Application website where students enter their learning objectives and area of interests.
- (2) Data pre-processing component where the student's entry is normalized.
- (3) Learning algorithm component where descriptions of courses are extracted using VSM and TF-IDF. A student  $S_a$  profile is learned using  $k$ NN algorithm. A list of top  $C$  courses is provided in  $L_c$ .
- (4) Recommendation engine component where the content-based filtering is used and similarity between courses and students is computed using cosine vector measure. The top  $C$  courses with highest similarity score are recommended.

### *Tools and Technical Component*

The libraries and tools used to develop PAARS web application are discussed in Table 6.2.

Table 6.2 Tools used to develop PAARS

Component	Library/Tool
UI	React
Feature Extraction	Scikit-learn, Rake-nltk
Text Processing	nltk, pandas, SpaCy
<i>k</i> NN Algorithm	Scikit-learn

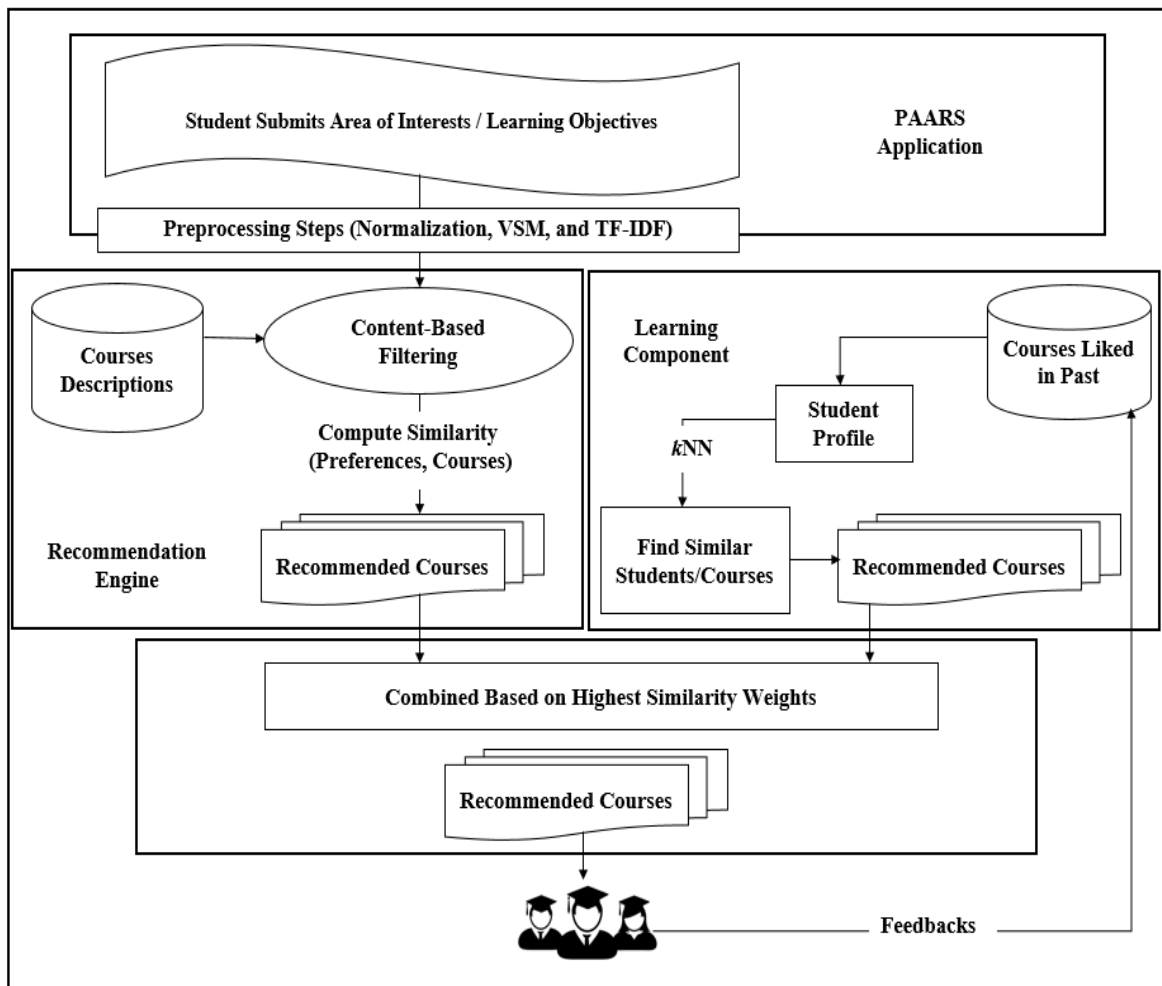


Figure 6.2 Framework architecture of PAARS web-based application

There are two datasets used to build the PAARS. The first dataset is courses and its descriptions (course name, course id, number of credits, department, and description). The total number of courses is 4834. The second dataset will be collected in the future after getting the approval from the Institutional Review Board at University of Idaho (UIIdaho). This dataset contains students' preferences after using PAARS. At this stage, our aim is to make PAARS available to get responses from UIIdaho students.

The dataset for courses is populated from the 2019-2020 UIIdaho Catalog<sup>1</sup>. The dataset includes courses for both undergraduate and graduate students. In the future, the scope of PAARS can be easily expanded to include other universities by simply storing their datasets.

### *Data Preprocessing*

This research involves an extensive data exploration, including visualization of frequently occurring words and phrases (unigrams, bigrams, and trigrams) [136]. Also, cluster analysis is performed to identify outliers in the course dataset, standardize the vocabularies, lowercase the descriptions of courses, lemmatize the descriptions of courses, remove stop-words, and normalize the descriptions of courses. It is noteworthy that preprocessing of the dataset is handled offline to optimize the performance of PAARS.

### *PAARS Web-Based Application*

The design of PAARS<sup>2</sup> is a user-friendly web-based application. Non-functional requirements are considered, for instance system's security, usability, performance and availability. The back-end of PAARS is implemented based on React framework to acquire the non-functional requirements mentioned above. It is noteworthy that PAARS application is hosted on Amazon Web Services which offer cloud web hosting solutions. Moreover, it is worth noting that PAARS is developed to conduct this research and is non-profit website.

On the client side, students can provide either their area of research interests or list of their learning objective. PAARS can handle either form of inputs. When students create an account and log in using their university email and password, they can submit their input which will be sent as a POST request to the server. The server accordingly orchestrates the entered input through the data preprocessing, learning component, and recommendation engine. The server responds back to students with a list of

---

<sup>1</sup> <https://catalog.uidaho.edu/>

<sup>2</sup> <http://3.136.194.233/>



recommended courses after the synchronous task orchestration is complete. The user-interfaces designs of PAARS web-based application are depicted in Figure 6.3 and Figure 6.4.

### Experiments and Evaluation

A total number of 200 students at UIIdaho, who have agreed to be volunteers and to take part in the experiment, will be asked to complete a survey to assess their satisfaction when using PAARS. The sample will be collected randomly by broadcasting an email to undergraduates and graduate students. The email will include the aim of the research study and encourage students to participate. Moreover, the email will include a survey based on a 5-point Likert scale (1: bad, to 5: excellent) to assess students' satisfaction when using PAARS. The questions will address whether they like the user interface of PAARS, how they feel about the complexity of the tasks, and the quality of the recommendation.

Since the dataset of the students' behaviors is not collected in advance in the first place, the user study and online experiments are used to store the interactions between students and PAARS and carried out to evaluate the quality of the recommendation. User study experiment is an important technique for evaluating RSs [137]. Moreover, online experiment can achieve the most real testing results among other evaluation methods. The main advantage of online experiment is that the entire performance of the PAARS will be evaluated. However, this step will be performed once the approval is received from the Institutional Review Board at UIIdaho.

Figure 6.3 The user interface for an entry of a student preferences

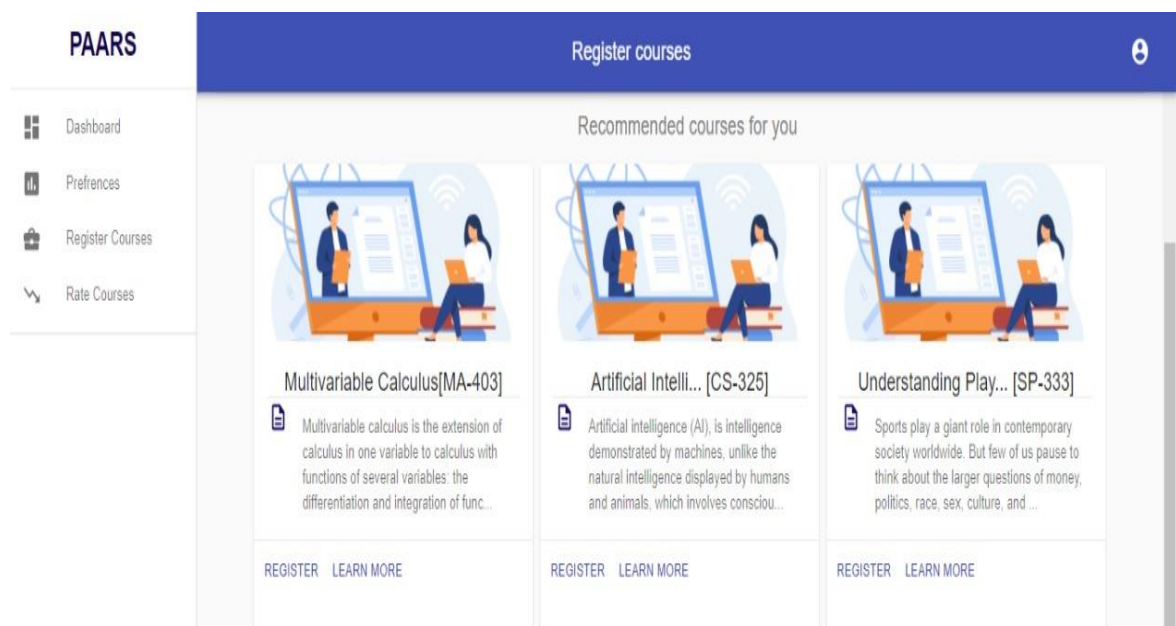


Figure 6.4 The user interface for a list of recommended courses based on a personalized preferences of a student

## Conclusion

In this research, an automated system PAARS is designed to help students in the process of courses selection based on their preferences. The main objective of PAARS is to mimic the role of academic advisors who help students take this hard decision by analyzing their preferences and personal profiles. The main advantages of PAARS are to help students find useful courses among a variety of available selection, to enhance their academic performance, and to improve their level of loyalty to their universities as well. Moreover, this system can be useful to advisors to find relevant courses that they might not be aware of and encourage students to enroll into these courses.

The proposed system is based on two primary techniques: (1) content-based filtering technique, and (2) an ensemble learning algorithm of kNN and VSM used to learn an active student profile and to suggest a list of courses based on a student's specific interests and other similar students' preferences. It considers that each student might prefer to enroll in a list of courses that correspond to a student's interests. The main input for this system is courses' description. Then the system updates each student's profile based on current and previous preferences. The output of the systems is a list of recommended courses. The mission of PAARS is to connect students and courses, which in one way helps students to find a list of courses valuable to them and in another way exposes the courses to specific students. This is the win-win situation for both students and advisors, and courses provided by the UIIdaho. Hence, PAARS is capable of providing guidance to students interested in pursuing their studies in the UIIdaho.

The design of PAARS is scalable and it can be expanded in the future to consider other universities in USA.

## Chapter 7: Fusing Social Network Influencers' Data to Augment Recommender Systems

*This chapter is based on:*

A. Althbiti and X. Ma, "Social Network Influencers' Data Augmenting Recommender Systems," *Proceedings of the 2020 International Conference on Computational Science and Computational Intelligence, Las Vegas, VA. In Press.*

### Introduction

Social network sites such as Facebook, Twitter, Instagram, and LinkedIn have gained and attracted billions of users [139]. Users use these platforms to interact with other people and to express their opinions regarding a specific topic. Each social network site aims to provide different services than other platforms. For instance, Facebook is introduced to connect college students with their colleagues and to build professional relationships [140]. Twitter is introduced so people could express their opinions in a restricted number of letters. These different platforms are developed to enhance the social-capital and to help build virtual communities that facilitate communication between users and friends.

The use of available data is what has motivated researchers to develop, examine and validate their models. For instance, researchers use these data in several studies in recommender systems [141], sentiment analysis [142], and marketing [143]. Social network site developers use these data to identify the similarity between users based on their beliefs, languages, or even demographic information [138]. Examining these similarities helps social network sites to understand users' needs and to predict new relationships between users.

Users can follow others who share similar beliefs, interests and preferences. Users can be influenced by other users' posts and comments [144]. In recommender systems, using relationships between users can augment the performance and the recommendation quality [83]. For instance, a model proposed in [145] evolves a recommender system that can search social network sites to identify and recommend learning peers to a user based on their posts, comments, and shared friends on the site. While researchers in [146] present a model that can determine a set of social network influencers by mining their posts, the present study introduces a model that integrates social network influencers' data to augment recommender systems. This model is based on three fundamental mechanisms: (1) the content-based technique, (2) a Probabilistic method, and (3) the influencers identifying technique.

The structure of this chapter is organized as follows. It first discusses a background of the conducted study. It then discusses the proposed model that leverages social network influencers' data to augment the performance of the recommender systems. It concludes by summarizing the main components of the proposed model.

### **Literature Review**

This section discusses the related topics that motivate this research. It discusses social network sites, recommender systems, and Bayesian classifier used to develop this model to learn an active user profile.

#### *Social Network Sites*

Researchers in [138] define social network sites as web-based services that let users:

- (1) build a public or semi-public user profile that might contain their interests and demographic information
- (2) provide current friendship networks and connections
- (3) identify new friendship connections

These connections between users can be identified in two different ways: (1) mutual connections, which means both users follow each other, or (2) one-direction link, which means a user is followed by other users whom he or she does not follow. The names for these links vary from site to site, though the common terms include "Friends," "Contacts," and "Fans" [138]. These relationships can be demonstrated as a network of nodes and links. The nodes signify users and edges signify their relationship [145].

#### *Recommender Systems*

Recommender systems are systems and software tools that automatically and effectively produce recommendations of the most suitable items to a target user by predicting user's interests and preferences [5]. The prediction process is based on the previous provided data of user's interests [6]. The aim of developing these systems is to reduce the overload of available selections by recommending relevant items to an active user. These systems provide personalized and un-personalized recommendations by developing models that analyze users' data, opinions and behaviors. In addition to analyzing users' data and opinions, they analyze user's data and opinions, but also analyze the opinions of other users who share similar preferences for augmenting the prediction accuracy.

The general scheme of recommender systems' mechanism is: (1) collect data about users and items, (2) compute the similarity between users and items, and (3) use the provided data to predict a rating of an unseen item in the future. The first part is done through any information retrieval technique such as

Keyword matching or the Vector Space Model [20]. The similarity can be computed by several affinity and similarity techniques such as Pearson Correlation Coefficients, or Cosine Vector. Equation (7.1) is the Pearson Correlation Coefficients used to compute the similarity between two users,

$$sim(u, v) = \frac{\sum_{i \in I} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I} (r_{u,i} - \bar{r}_u)^2} * \sqrt{\sum_{i \in I} (r_{v,i} - \bar{r}_v)^2}} \quad (7.1)$$

where  $\bar{r}_u$  and  $\bar{r}_v$  are the average rating of the available ratings provided by users  $u$  and  $v$ , and  $i \in I$  means these items that user  $u$  and  $v$  previously both rated.

### *Bayesian Classifier*

A Bayesian classifier is a supervised machine learning algorithm, based on probabilistic framework, used to solve a classification problem. Although this technique is quite similar to the linear regression models, it tends to be even faster in the training step [38]. It is used to address the latent characteristics of the relationships learned from the data. It models these relationships based on three fundamental concepts: (1) prior, (2) posterior, and (3) likelihood. The prior concept forms the prior knowledge about what the expected relationship looks like. The knowledge in the prior is driven and inferred from the current relationships in the training dataset. Moreover, the probability of a model “posterior” is proportional to the product of the likelihood times the prior probability of a class. The likelihood function measures the goodness of fit of a predictive model to a sample of data for specified values of the unidentified features.

### **Proposed Approach**

There are three fundamental mechanisms used to build this model: (1) content-based technique, (2) probabilistic method, and (3) algorithm to identify social media influencers [146]. The methodology of incorporating influencers’ data to augment recommender systems is as follows. The first step is to determine a set of influencers  $U$  from social network sites, where  $U=1,2, \dots, n$ . Researchers in [146] discuss an algorithm to identify a set of social network influencers by mining their posts. They discuss that identifying those influencers is done through three main steps: (1) the number of tweets provided by a user, (2) the number of mentions by other users for a user, and (3) the number of retweets for a user’s posts.

The second step is to stream and use their data from Twitter or other social network sites that allow the streaming process such as Facebook. Keyword matching or the Vector Space Model can be used as retrieval models [20]. These collected posts or tags are analyzed by Tweets Analyzer built based on an information retrieval technique. The unstructured retrieved data of each influencer  $U_n$  is mapped to

construct a vector  $V_{Un}$  that contains structured data of posts and tags. The Extracted Items component stores these vectors for each influencer in a structured form.

The second component is Item Filtering component, which performs the following: (1) compute the similarity between an active user profile, from the Updated Profiles component and extracted influencers, and (2) filter influencers whose data is used to compute the recommendation of an item for an active user. It is noteworthy that a profile of an active user is also defined as a vector  $V_{An}$ . The term frequency-inverse document frequency weighting scheme is used to weight the tags and posts of both an active user and an influencer. Figure 7.1 demonstrates the prototype of the proposed model.

The third algorithm used to learn an active user's profile is Naïve Bayes, a class of Bayesian classifiers [20]. A Bayesian classifier is a supervised machine learning algorithm, based on probabilistic framework, used to solve a classification problem. Given a vector or a user profile of an active user with  $N$  words ( $I1, I2, I3, \dots, I_N$ ), the aim is to find the relevant influencers. Estimating a probability( $V_{Un} | I1, I2, I3, \dots, I_N$ ) is used to determine the affinity between an active user and an influencer  $V_{Un}$ . Next, a Bayes' theorem (7.2) is applied,

$$P(V_{Un}|I1, I2, I3, \dots, I_N) \propto P(I1, I2, I3, \dots, I_N | V_{Un}) * P(V_{Un}) \quad (7.2)$$

Using a common Bayesian classifier known as Naïve Bayes classifier, the conditional probability of  $P(I1, I2, I3, \dots, I_N | V_{Un})$  can be estimated by (7.3),

$$(I1|V_{Un}) * P(I2|V_{Un}) * P(I3|V_{Un}) \dots P(I_N|V_{Un}) \quad (7.3)$$

Researchers [145] discuss that Naïve Bayes classifier outperforms other classifiers in making good recommendations. They also mention that the common use of Naïve Bayes classifier due to its ability to add prior knowledge, good prediction time, and its ability to prevent that presence or absence of one of those words  $I1$  in  $V_{Un}$  will not affect the other.

Then, the model of multinomial Naïve Bayes is used to calculate  $P(I1|V_{Un})$ . Equation (7.4) is used to count how many times a word  $I_N$  appeared in  $V_{Un}$ ,

$$P(I1|V_{Un}) = P(I_N) \prod_{t_k \in V_{Un}} P(t_k|I_N)^{N(V_{Un}, t_k)} \quad (7.4)$$

where  $N(V_{Un}, t_k)$  is defined as the number of times a word  $t_k$  appeared in  $V_{Un}$ . To identify influencers whose data can be incorporated in the prediction process, the probability obtained from their respective vectors is sorted.

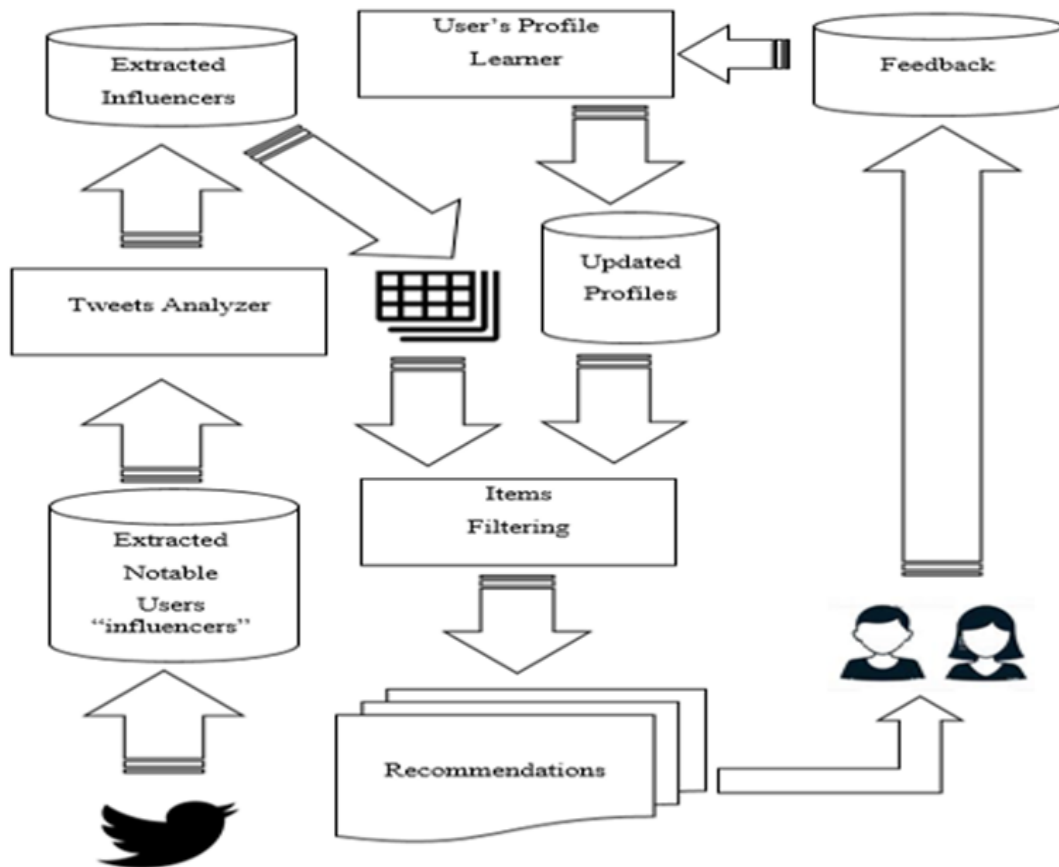


Figure 7.1 The prototype of the model that fuses information from Social Network Sites

## Conclusion

To augment the quality of the recommendations, this chapter introduces a model that incorporates and integrates social network influencers' data to enhance the prediction process in recommender systems. This model is based on three fundamental techniques: (1) content-based recommender system, (2) Bayesian classifier, and (3) influencers' identifying technique. The technique used to identify influencers in social network sites is based on three main steps: (1) the number of tweets provided by a user, (2) the number of mentions by other users for a user, and (3) the number of retweets for a user's posts. The proposed model uses Naïve Bayes classifier to determine influencers whose data can be used in the prediction process for unseen items of an active user.



## **Chapter 8: Conclusions, Limitations, and Future Research Directions**

### **Introduction**

This empirical research addresses the main drawbacks in the field of collaborative filtering (CF). This chapter begins by summarizing the work contained in this experimental dissertation, outlining the main developments, and discussing the primary results. Next it considers the practicability of scaling one of the proposed models to develop a personalized academic advisory recommender system. Finally, it addresses the limitations of this dissertation and points out interesting directions for future research.

### **Summary**

The preceding chapters of this dissertation present four models that address the data sparsity and limited coverage issues in CF. This empirical study discusses detailed descriptions for a variety of proposed models in CF, demonstrates their relationship to machine learning (ML) algorithms, and analyzes their performance in terms of rating prediction, ranking accuracy, and space requirements.

It argues that the main drawbacks and limitations in the conventional CF systems are the data sparsity and the limited coverage. The data sparsity issue means that the user-item rating matrix is not fully specified, which leads to unreasonable recommendations and to poor prediction performance. This often happens because not all users are able to rate all the available items in a CF system.

The limited coverage issue means that when calculating the similarity between users or items, only users who rate the same items are considered as neighbors to a target user, and only items rated by the same users are considered as neighbors to a target item. It is noteworthy that several researchers use other terms to explain the issue of limited coverage, for instance “collaboratively rated” or “co-rated” items. Therefore, this study begins by discussing the potential needs of CF, research related to the conventional CF methods and to its uses, CF applications, popular datasets in CF, CF evaluation metric, and CF limitations.

Chapter 2 outlines the potential needs to develop algorithms to address the challenge of selection overload. It claims that CF systems have been developed to help users to determine which of the available selections interest them. CF systems have been widely discussed as an efficient approach to cope with the selection overload issue. Chapter 2 presents a literature review of the CF techniques and data mining algorithms used for CF. Moreover, it discusses the popular datasets and evaluation metrics used to assess the performance of CF.

Chapter 3 discusses advanced ML algorithms that intrinsically form the model-based CF systems. It also states that ML addresses the current research needs of how to build models that learn automatically through experience. Chapter 3 reviews supervised, unsupervised, and reinforcement learning and discusses dimensionality reduction techniques including singular value decomposition and principal components analysis. Moreover, it discusses how ML algorithms can alleviate the issue of dimensionality and offer solutions to automate prediction and detection. This chapter delineates how the emergence of learning approaches and the explosion of big data motivate the recent development of ML models.

The use of clustering and artificial neural network is presented in chapter 4. Chapter 4 discusses a proposed model that addresses the data sparsity issue in CF. It proves the novelty of the proposed model by comparing its accuracy results with other CF algorithms. It intensively conducts a set of experiments on four popular data sets to assess the performance of the proposed model. The results prove that integrating clustering with artificial neural network effectively addresses the sparsity issue, improves the quality of recommendations, and demonstrates promising prediction accuracy. Therefore, this chapter addresses research questions 1 and 2.

Chapter 5 discusses the use of rating alignment-based similarity model in CF to address the issues of data sparsity and limited coverage. In order to improve the accuracy and the recommendation quality of a CF system, this chapter proposes a novel similarity model that identifies regions of similarity between a pair of users in the rating matrix to compute the similarity between users. Based on these similar regions, CF can use the similarity scores between users and items to predict an active user's ratings on unseen items. The similarity model is novel in two respects: (1) the prediction process is not sensitive to the number of  $k$ -nearest neighbors, and (2) the similarity computation considers both the collaboratively rated (co-rated) items to compute the similarity scores in relatively sparse datasets and the alignment of the un-co-rated items of users to classify relevant neighborhoods and generate recommendation. This chapter addresses research questions 3 and 4.

Examining the feasibility of the proposed models is one of these research objectives. Chapter 6 develops a personalized academic advisory based on CF. It begins the discussion by stating the limitations of the conventional advisory task when recommending a list of potential courses to the students at University of Idaho. The main limitations of the conventional advisory task are time consuming, risky, and monotonous work. These issues negatively affects students' performance and learning experience. The data mining technique used to learn profiles for students is a  $k$ -nearest neighbors ( $k$ NN) classifier. The

developed system shows the practicability of CF. Hence, research question 5 is addressed in this chapter.

Incorporating auxiliary information from social network platforms is a cutting-edge technology that attracts CF researchers. Chapter 7 demonstrates the essential use of incorporating data from external sources. It states that the ever-increasing use of social network sites and the availability of Internet services have introduced opportunities for researchers to augment the performance predictions of CF. Chapter 7 proposes a novel CF hybrid model based on three main techniques: (1) a content-based technique, (2) a Bayesian classifier, and (3) a model to identify social media influencers. The proposed model integrates CF with these three models to augment the prediction performance. This chapter addresses research question 6.

### **Limitations**

Every study has limitations related to aspects such as availability of datasets or computation capability. Since the results of this dissertation's experiments demonstrate a more accurate performance and a higher-quality recommendation, the limitations are not related to the novelty aspects of the proposed models.

The first limitation is related to finding a useful dataset that includes popular items that allow users to review items on social network sites. When users provide ratings, they are encouraged to do so directly to a CF system. It is uncommon for users to use social network sites to provide ratings of their daily purchases, for instance. Hence, obtaining a user-item dataset from social network sites is a real challenge and limitation of this study.

The second limitation is extended to conducting a user study evaluation of the developed CF system for the students at University of Idaho. This type of evaluation requires a real interaction between a developer and users to assess users' reactions when using a CF system. Because of health considerations related to the current COVID-19 pandemic, the evaluation for the developed system in chapter 6 is postponed and kept for future work.

### **Future Work**

This section considers possible directions for future work. It discusses employing some of the introduced models in more advanced CF. It discusses the work left to be performed with proposed models. Lastly, it points out other application domains where the proposed models can be used.

### *Extending the Presented Models*

Although; this dissertation offers a fairly comprehensive evaluation of a set of proposed models, some additional work with these models needs to be studied. Incorporating clustering and artificial neural network with principal component analysis (PCA) addresses the data sparsity problem, as discussed in chapter 4. However, this hybrid model has not been used to examine its performance when addressing the cold-start issue. The issue of cold-start is popular in the domain of CF. It appears when there are no ratings provided at all for a new user or item. It moreover arises when a system is just recently developed with no preference information. Therefore, this study indicates that there is a potential research gap to address the issue of cold-start in CF. Experimentation with the clustering and artificial neural network model could be performed to determine its ability to address this issue.

Another discussed model is related to introducing a similarity model to be integrated with a conventional CF to address the limited coverage issue, as discussed in chapter 5. The proposed similarity model finds similar regions between users in the rating matrix after performing rating alignment task given a set of conditions. These conditions are associated with costs. For instance, if there are two similar ratings between two users, this similarity is called match and is given a specific score. Similarly, if two ratings are not similar, it is called mismatch and is given a specific score. Then, the model computes the total similarity scores between users. The proposed model here does consider all the provided ratings in the user-item matrix, while other similarity models only consider co-rated items. The performance of the proposed model has been evaluated on four datasets. However, considering more datasets with high sparsity ratio is a current research gap that needs to be addressed. Also, examining the main characteristics of the proposed similarity models is an interested research direction. This model is not sensitive to the  $k$ -number of nearest neighbors. It only requires a number of  $k$  to be 2, unlike other similarity models that require the number of the nearest neighbors to range from 20 to 50. Therefore, intensive experiments on the effect of dataset size, high ratio of the sparsity, number of neighbors, and optimizing the cost function are clearly needed. A comprehensive understanding of how all these components would develop a model that overcomes both data sparsity and limited coverage issues and achieve better prediction accuracy.

### *Continuing the Left Work*

As discussed previously in the limitation section, there are a few components left to be done in the future. These components are not essential at this time and are not functional requirements. However, these components can form another study. For instance, finding appropriate datasets that include information about social network influencers and their preferences related to a specific domain such as

music or movie is needed to examine the performance of the proposed model in chapter 7. To recall the discussion about the proposed model in chapter 7, it is developed to incorporate auxiliary information from external sources to augment CF. The information includes a set of social network influencers and their opinions about items and how their preferences could influence other users.

Another research gap is to conduct statistical analysis and user-study experiments to evaluate the usefulness, ease of use, and the validity of the results of the Personalized Academic Advisory Recommender System, as discussed in chapter 6.

### **Last Word**

The main motivation of this research is to augment the performance of CF. This experimental study is conducted to address the data sparsity and limited coverage issues in CF. Models are evaluated on real datasets collected from popular domains. The results of the intensive experiments reveal that the proposed models are accurate and outperform other current CF algorithms.

## Reference Cited

- [1] F. Ricci, L. Rokach, and B. Shapira, "Introduction to Recommender Systems Handbook," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Boston, MA: Springer US, 2015, pp. 1–34.
- [2] N. J. Belkin and W. B. Croft, "Information Filtering and Information Retrieval: Two Sides of the Same Coin," *Communications of the Acm*, vol. 35, no. 12, pp. 29–38, 1992.
- [3] C. C. Aggarwal, *Recommender Systems*. Cham: Springer International Publishing, 2016.
- [4] A. S. Lampropoulos and G. Tsihrantzēs, Eds., *Machine learning paradigms: applications in recommender systems*. Cham: Springer, 2015.
- [5] P. Melville and V. Sindhwani, "Recommender Systems," in *Encyclopedia of Machine Learning and Data Mining*, C. Sammut and G. I. Webb, Eds. Boston, MA: Springer US, 2017, pp. 1056–1066.
- [6] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural Collaborative Filtering," in *Proceedings of the 26th International Conference on World Wide Web*, Republic and Canton of Geneva, Switzerland, 2017, pp. 173–182.
- [7] A. Halavais, *Search engine society*, Second edition. Cambridge, UK Medford, MA, USA: Polity, 2018.
- [8] U. Shardanand and P. Maes, "Social Information Filtering: Algorithms for Automating 'Word of Mouth,'" in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, New York, NY, USA, 1995, pp. 210–217.
- [9] X. Ning, C. Desrosiers, and G. Karypis, "A Comprehensive Survey of Neighborhood-Based Recommendation Methods," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 37–76.
- [10] A. Hernando, J. Bobadilla, and F. Ortega, "A Non Negative Matrix Factorization for Collaborative Filtering Recommender Systems Based on a Bayesian Probabilistic Model," *Know.-Based Syst.*, vol. 97, no. C, pp. 188–202, Apr. 2016.
- [11] P. Valdiviezo-Diaz, F. Ortega, E. Cobos, and R. Lara-Cabrera, "A Collaborative Filtering Approach Based on Naïve Bayes Classifier," *IEEE Access*, vol. 7, pp. 108581–108592, 2019.

- [12] M. D. Ekstrand, “Collaborative Filtering Recommender Systems,” *FNT in Human–Computer Interaction*, vol. 4, no. 2, pp. 81–173, 2011.
- [13] X. Zhou and S. Wu, “Rating LDA Model for Collaborative Filtering,” *Know.-Based Syst.*, vol. 110, no. C, pp. 135–143, Oct. 2016.
- [14] L. Huang, W. Tan, and Y. Sun, “Collaborative recommendation algorithm based on probabilistic matrix factorization in probabilistic latent semantic analysis,” *Multimed Tools Appl*, vol. 78, no. 7, pp. 8711–8722, Apr. 2019.
- [15] Y. Koren, “Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model,” in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2008, pp. 426–434.
- [16] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, “Variational Autoencoders for Collaborative Filtering,” in *Proceedings of the 2018 World Wide Web Conference*, Republic and Canton of Geneva, Switzerland, 2018, pp. 689–698.
- [17] B. Abdollahi and O. Nasraoui, “Explainable Restricted Boltzmann Machines for Collaborative Filtering,” *arXiv:1606.07129 [cs, stat]*, Jun. 2016.
- [18] D. Bokde, S. Girase, and D. Mukhopadhyay, “Matrix Factorization Model in Collaborative Filtering Algorithms: A Survey,” *Procedia Computer Science*, vol. 49, pp. 136–146, 2015.
- [19] M. Nilashi, O. bin Ibrahim, N. Ithnin, and N. H. Sarmin, “A multi-criteria collaborative filtering recommender system for the tourism domain using Expectation Maximization (EM) and PCA–ANFIS,” *Electronic Commerce Research and Applications*, vol. 14, no. 6, pp. 542–562, Oct. 2015.
- [20] M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro, “Semantics-Aware Content-Based Recommender Systems,” in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 119–159.
- [21] G. Li and W. Ou, “Pairwise probabilistic matrix factorization for implicit feedback collaborative filtering,” *Neurocomputing*, vol. 204, pp. 17–25, Sep. 2016.
- [22] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245, pp. 255–260, Jul. 2015.
- [23] Y. Koren, R. Bell, and C. Volinsky, “Matrix Factorization Techniques for Recommender Systems,” *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
- [24] J. B. Schafer, J. Konstan, and J. Riedi, “Recommender systems in e-commerce,” in *Proceedings of the 1st ACM conference on Electronic commerce - EC '99*, Denver, Colorado, United States, 1999, pp. 158–166.

- [25] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou, "Mobile recommender systems in tourism," *Journal of Network and Computer Applications*, vol. 39, pp. 319–333, Mar. 2014.
- [26] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Scalable Collaborative Filtering Approaches for Large Recommender Systems," *Journal of Machine Learning Research*, vol. 10, no. Mar, pp. 623–656, 2009.
- [27] B. M. Sarwar, J. A. Konstan, A. Borchers, J. Herlocker, B. Miller, and J. Riedl, "Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System," in *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work*, New York, NY, USA, 1998, pp. 345–354.
- [28] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl, "PolyLens: A Recommender System for Groups of Users," in *ECSCW 2001*, W. Prinz, M. Jarke, Y. Rogers, K. Schmidt, and V. Wulf, Eds. Dordrecht: Kluwer Academic Publishers, 2002, pp. 199–218.
- [29] B. Xu, J. Bu, C. Chen, and D. Cai, "An Exploration of Improving Collaborative Recommender Systems via User-item Subgroups," in *Proceedings of the 21st International Conference on World Wide Web*, New York, NY, USA, 2012, pp. 21–30.
- [30] A. S. Das, M. Datar, A. Garg, and S. Rajaram, "Google news personalization: scalable online collaborative filtering," in *Proceedings of the 16th international conference on World Wide Web - WWW '07*, Banff, Alberta, Canada, 2007, p. 271.
- [31] A. Felfernig and R. Burke, "Constraint-based Recommender Systems: Technologies and Research Issues," in *Proceedings of the 10th International Conference on Electronic Commerce*, New York, NY, USA, 2008, pp. 3:1–3:10.
- [32] J. Davidson *et al.*, "The YouTube Video Recommendation System," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, New York, NY, USA, 2010, pp. 293–296.
- [33] N. T. Thong and L. H. Son, "HIFCF: An effective hybrid model between picture fuzzy clustering and intuitionistic fuzzy recommender systems for medical diagnosis," *Expert Systems with Applications*, vol. 42, no. 7, pp. 3682–3701, May 2015.
- [34] P. Paranjape-Voditel and U. Deshpande, "A stock market portfolio recommender system based on association rule mining," *Applied Soft Computing*, vol. 13, no. 2, pp. 1055–1063, Feb. 2013.
- [35] H. Drachsler, K. Verbert, O. C. Santos, and N. Manouselis, "Panorama of Recommender Systems to Support Learning," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 421–451.



- [36] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: A survey," *Decision Support Systems*, vol. 74, pp. 12–32, Jun. 2015.
- [37] Z. Yang, B. Wu, K. Zheng, X. Wang, and L. Lei, "A Survey of Collaborative Filtering-Based Recommender Systems for Mobile Internet Applications," *IEEE Access*, vol. 4, pp. 3273–3287, 2016.
- [38] X. Amatriain, A. Jaimes\*, N. Oliver, and J. M. Pujol, "Data Mining Methods for Recommender Systems," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Boston, MA: Springer US, 2015, pp. 227–262.
- [39] M. K. Najafabadi, A. Hj. Mohamed, and M. N. Mahrin, "A survey on data mining techniques in recommender systems," *Soft Comput*, vol. 23, no. 2, pp. 627–654, Jan. 2019.
- [40] Y. Koren and R. Bell, "Advances in Collaborative Filtering," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 77–118.
- [41] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders Meet Collaborative Filtering," in *Proceedings of the 24th International Conference on World Wide Web*, New York, NY, USA, 2015, pp. 111–112.
- [42] H. Zhang, F. Shen, W. Liu, X. He, H. Luan, and T.-S. Chua, "Discrete Collaborative Filtering," in *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2016, pp. 325–334.
- [43] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 5–53, Jan. 2004.
- [44] P. B. Thorat, R. M. Goudar, and S. Barve, *Survey on Collaborative Filtering, Content-based Filtering and Hybrid Recommendation System*.
- [45] F. Braida, C. E. Mello, M. B. Pasinato, and G. Zimbrão, "Transforming Collaborative Filtering into Supervised Learning," *Expert Syst. Appl.*, vol. 42, no. 10, pp. 4733–4742, Jun. 2015.
- [46] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, Mar. 1970, doi: 10.1016/0022-2836(70)90057-4.
- [47] A. Bilge and C. Kaleli, "A multi-criteria item-based collaborative filtering framework," in *2014 11th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 2014, pp. 18–22.
- [48] H. Koochi and K. Kiani, "User based Collaborative Filtering using fuzzy C-means," *Measurement*, vol. 91, pp. 134–139, Sep. 2016.

- [49] S. Ghazarian and M. A. Nematbakhsh, "Enhancing memory-based collaborative filtering for group recommender systems," *Expert Systems with Applications*, vol. 42, no. 7, pp. 3801–3812, May 2015.
- [50] G. Guo, J. Zhang, and N. Yorke-Smith, "Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems," *Knowledge-Based Systems*, vol. 74, pp. 14–27, Jan. 2015.
- [51] M. M. Azadjalal, P. Moradi, A. Abdollahpouri, and M. Jalili, "A trust-aware recommendation method based on Pareto dominance and confidence concepts," *Knowledge-Based Systems*, vol. 116, pp. 130–143, Jan. 2017.
- [52] M. Kaminskis and D. Bridge, "Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems," *ACM Trans. Interact. Intell. Syst.*, vol. 7, no. 1, pp. 1–42, Dec. 2016.
- [53] Y. Leng, Q. Lu, and C. Liang, "Black-start decision making based on collaborative filtering for power system restoration," *International Journal of Electrical Power & Energy Systems*, vol. 100, pp. 279–286, Sep. 2018.
- [54] W. Gong, L. Qi, and Y. Xu, "Privacy-Aware Multidimensional Mobile Service Quality Prediction and Recommendation in Distributed Fog Environment," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1–8, 2018.
- [55] H. Parvin, P. Moradi, and S. Esmaili, "TCFACO: Trust-aware collaborative filtering method based on ant colony optimization," *Expert Systems with Applications*, vol. 118, pp. 152–168, Mar. 2019.
- [56] I. Fernández-Tobías, M. Braunhofer, M. Elahi, F. Ricci, and I. Cantador, "Alleviating the New User Problem in Collaborative Filtering by Exploiting Personality Information," *User Modeling and User-Adapted Interaction*, vol. 26, no. 2–3, pp. 221–255, Jun. 2016.
- [57] Burke, R., O'Mahony, M. P., & Hurley, N. J., "Robust collaborative recommendation." in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 961–995.
- [58] D. Goldberg, D. Nichols, B.M. Oki, D. Terry, Using collaborative filtering to weave an information tapestry, *Communications of the ACM*, 35 (1992) 61-70.
- [59] R. Chen, Q. Hua, Y.-S. Chang, B. Wang, L. Zhang, and X. Kong, "A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks," *IEEE Access*, vol. 6, pp. 64301–64320, 2018.

- [60] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '99*, Berkeley, California, United States, 1999, pp. 230–237.
- [61] C. C. Aggarwal, *Data Mining*. Cham: Springer International Publishing, 2015.
- [62] M. Amadeo, C. Campolo, and A. Molinaro, "Multi-source Data Retrieval in IoT via Named Data Networking," in *Proceedings of the 1st ACM Conference on Information-Centric Networking*, New York, NY, USA, 2014, pp. 67–76.
- [63] C. Zhai, "Probabilistic Topic Models for Text Data Retrieval and Analysis," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2017, pp. 1399–1401.
- [64] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation and collaborative filtering," in *Proceedings of the fourth ACM conference on Recommender systems - RecSys '10*, Barcelona, Spain, 2010, p. 119.
- [65] S. M. McNee, J. Riedl, and J. A. Konstan, "Being accurate is not enough: how accuracy metrics have hurt recommender systems," in *CHI '06 extended abstracts on Human factors in computing systems - CHI EA '06*, Montré#233;al, Qu#233;bec, Canada, 2006, p. 1097.
- [66] D. Lian *et al.*, "Scalable Content-Aware Collaborative Filtering for Location Recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1122–1135, Jun. 2018.
- [67] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261–273, Nov. 2015.
- [68] J. Lian, F. Zhang, X. Xie, and G. Sun, "CCCFNet: A Content-Boosted Collaborative Filtering Neural Network for Cross Domain Recommender Systems," in *Proceedings of the 26th International Conference on World Wide Web Companion*, Republic and Canton of Geneva, Switzerland, 2017, pp. 817–818.
- [69] D. Agarwal, B.-C. Chen, and B. Long, "Localized factor models for multi-context recommendation," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining -KDD '11*, San Diego, California, USA, 2011, p. 609.
- [70] R. P. Adams, G. E. Dahl, and I. Murray, "Incorporating Side Information in Probabilistic Matrix Factorization with Gaussian Processes," *arXiv:1408.2039 [cs, stat]*, Aug. 2014.
- [71] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J Royal Statistical Soc B*, vol. 67, no. 2, pp. 301–320, Apr. 2005.

- [72] I. Guy, “Social Recommender Systems,” in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 511–543.
- [73] M. Schedl, P. Knees, B. McFee, D. Bogdanov, and M. Kaminskas, “Music Recommender Systems,” in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 453–492.
- [74] E. Varga, “Recommender Systems,” in *Practical Data Science with Python 3*, Berkeley, CA: Apress, 2019, pp. 317–339.
- [75] F. Garcin, B. Faltings, O. Donatsch, A. Alazzawi, C. Bruttin, and A. Huber, “Offline and Online Evaluation of News Recommender Systems at Swissinfo.Ch,” in *Proceedings of the 8th ACM Conference on Recommender Systems*, New York, NY, USA, 2014, pp. 169–176.
- [76] X. Dong, L. Yu, Z. Wu, Y. Sun, L. Yuan, and F. Zhang, “A Hybrid Collaborative Filtering Model with Deep Structure for Recommender Systems,” in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, California, USA, 2017, pp. 1309–1315.
- [77] B. Smith and G. Linden, “Two Decades of Recommender Systems at Amazon.com,” *IEEE Internet Computing*, vol. 21, no. 3, pp. 12–18, May 2017.
- [78] X. Li, X. Cheng, S. Su, S. Li, and J. Yang, “A hybrid collaborative filtering model for social influence prediction in event-based social networks,” *Neurocomputing*, vol. 230, pp. 197–209, Mar. 2017, doi: 10.1016/j.neucom.2016.12.024.
- [79] S. Jiang, X. Qian, J. Shen, and T. Mei, “Travel Recommendation via Author Topic Model Based Collaborative Filtering,” in *MultiMedia Modeling*, vol. 8936, X. He, S. Luo, D. Tao, C. Xu, J. Yang, and M. A. Hasan, Eds. Cham: Springer International Publishing, 2015, pp. 392–402.
- [80] X. Ning and G. Karypis, “Sparse linear methods with side information for top-n recommendations,” in *Proceedings of the sixth ACM conference on Recommender systems - RecSys '12*, Dublin, Ireland, 2012, p. 155.
- [81] M. Nilashi, O. Ibrahim, and K. Bagherifard, “A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques,” p. 3.
- [82] C. C. Aggarwal and S. Parthasarathy, “Mining massively incomplete data sets by conceptual reconstruction,” in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '01*, San Francisco, California, 2001, pp. 227–232.
- [83] T. Ma *et al.*, “Social Network and Tag Sources Based Augmenting Collaborative Recommender System,” *IEICE Trans. Inf. & Syst.*, vol. E98.D, no. 4, pp. 902–910, 2015.

- [84] P. Bachman, A. Sordoni, and A. Trischler, “Learning Algorithms for Active Learning,” in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, Sydney, NSW, Australia, 2017, pp. 301–310.
- [85] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.
- [86] S. Parthasarathy and C. C. Aggarwal, “On the use of conceptual reconstruction for mining massively incomplete data sets,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 6, pp. 1512–1521, Nov. 2003.
- [87] C. Musto, F. Narducci, P. Lops, M. De Gemmis, and G. Semeraro, “ExpLOD: A Framework for Explaining Recommendations Based on the Linked Open Data Cloud,” in *Proceedings of the 10th ACM Conference on Recommender Systems*, New York, NY, USA, 2016, pp. 151–154.
- [88] R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data*, 2nd ed. Hoboken, N.J: Wiley, 2002.
- [89] D. Jannach, Ed., *Recommender systems: an introduction*. New York: Cambridge University Press, 2011.
- [90] A. Friedman, S. Berkovsky, and M. A. Kaafar, “A differential privacy framework for matrix factorization recommender systems,” *User Model User-Adap Inter*, vol. 26, no. 5, pp. 425–458, Dec. 2016.
- [91] C. Zhang, J. Liu, Y. Qu, T. Han, X. Ge, and A. Zeng, “Enhancing the robustness of recommender systems against spammers,” *PLoS One*, vol. 13, no. 11, Nov. 2018.
- [92] M. Y. H. Al-Shamri, “User profiling approaches for demographic recommender systems,” *Knowledge-Based Systems*, vol. 100, pp. 175–187, May 2016.
- [93] D. Shriver, S. G. Elbaum, M. B. Dwyer, and D. S. Rosenblum, “Evaluating Recommender System Stability with Influence-Guided Fuzzing,” in *AAAI*, 2019.
- [94] A. Said and V. Torra, *Data science in practice*. 2019.
- [95] M. Erdt, A. Fernández, and C. Rensing, “Evaluating Recommender Systems for Technology Enhanced Learning: A Quantitative Survey,” *IEEE Transactions on Learning Technologies*, vol. 8, no. 4, pp. 326–344, Oct. 2015.
- [96] F. Strub, J. Mary, and R. Gaudel, “Hybrid Collaborative Filtering with Autoencoders,” *arXiv:1603.00806 [cs]*, Mar. 2016.

- [97] S. M. McNee, J. Riedl, and J. A. Konstan, "Making recommendations better: an analytic model for human-recommender interaction," in *CHI '06 extended abstracts on Human factors in computing systems - CHI EA '06*, Montréal, Québec, Canada, 2006, p. 1103.
- [98] R. D. Burke, H. Abdollahpouri, B. Mobasher, and T. Gupta, "Towards Multi-Stakeholder Utility Evaluation of Recommender Systems," in *UMAP*, 2016.
- [99] R. Bell, Y. Koren, and C. Volinsky, "Modeling relationships at multiple scales to improve accuracy of large recommender systems," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '07*, San Jose, California, USA, 2007, p. 95.
- [100] J. Lee, C. Jun, J. Lee, and S. Kim, "Classification-based collaborative filtering using market basket data," *Expert Systems with Applications*, vol. 29, no. 3, pp. 700–704, Oct. 2005.
- [101] S. Zhang, L. Yao, and X. Xu, "AutoSVD++: An Efficient Hybrid Collaborative Filtering Model via Contractive Auto-encoders," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2017, pp. 957–960.
- [102] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A Constant Time Collaborative Filtering Algorithm," *Information Retrieval*, vol. 4, no. 2, pp. 133–151, 2001.
- [103] B. Shams and S. Haratizadeh, "Graph-based collaborative ranking," *Expert Systems with Applications*, vol. 67, pp. 59–70, Jan. 2017.
- [104] T. Hastie, J. H. Friedman, and R. Tibshirani, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction." .
- [105] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015, doi: 10.1038/nature14236.
- [106] R. R. Picard and R. D. Cook, "Cross-Validation of Regression Models," *Journal of the American Statistical Association*, vol. 79, no. 387, pp. 575–583, Sep. 1984, doi: 10.1080/01621459.1984.10478083.
- [107] J. Wang, W. Liu, S. Kumar, and S.-F. Chang, "Learning to Hash for Indexing Big Data—A Survey," *Proc. IEEE*, vol. 104, no. 1, pp. 34–57, Jan. 2016, doi: 10.1109/JPROC.2015.2487976.
- [108] M. K. Najafabadi and M. N. Mahrin, "A systematic literature review on the state of research and practice of collaborative filtering technique and implicit feedback," *Artif Intell Rev*, vol. 45, no. 2, pp. 167–201, Feb. 2016.
- [109] A. Althbiti and X. Ma, "Collaborative Filtering," in *Encyclopedia of Big Data*, L. A. Schintler and C. L. McNeely, Eds. Cham: Springer International Publishing, 2019, pp. 1–4.

- [110] S. Natarajan, S. Vairavasundaram, S. Natarajan, and A. H. Gandomi, "Resolving data sparsity and cold start problem in collaborative filtering recommender system using Linked Open Data," *Expert Systems with Applications*, vol. 149, p. 113248, Jul. 2020, doi: 10.1016/j.eswa.2020.113248.
- [111] Y. Du, C. Yao, S. Huo, and J. Liu, "A new item-based deep network structure using a restricted Boltzmann machine for collaborative filtering," *Frontiers Inf Technol Electronic Eng*, vol. 18, no. 5, pp. 658–666, May 2017, doi: 10.1631/FITEE.1601732.
- [112] Z. Huang, H. Chen, and D. Zeng, "Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 116–142, Jan. 2004, doi: 10.1145/963770.963775.
- [113] X. Yang, S. Zhou, and M. Cao, "An Approach to Alleviate the Sparsity Problem of Hybrid Collaborative Filtering Based Recommendations: The Product-Attribute Perspective from User Reviews," *Mobile Netw Appl*, vol. 25, no. 2, pp. 376–390, Apr. 2020, doi: 10.1007/s11036-019-01246-2.
- [114] J. Lee, H.-J. Kim, and S. Lee, "Conceptual collaborative filtering recommendation: A probabilistic learning approach," *Neurocomputing*, vol. 73, no. 13–15, pp. 2793–2796, Aug. 2010, doi: 10.1016/j.neucom.2010.04.005.
- [115] B. Ait Hammou and A. Ait Lahcen, "FRAIPA: A fast recommendation approach with improved prediction accuracy," *Expert Systems with Applications*, vol. 87, pp. 90–97, Nov. 2017, doi: 10.1016/j.eswa.2017.06.001.
- [116] I. Fernández-Tobías, I. Cantador, P. Tomeo, V. W. Anelli, and T. Di Noia, "Addressing the user cold start with cross-domain collaborative filtering: exploiting item metadata in matrix factorization," *User Model User-Adap Inter*, vol. 29, no. 2, pp. 443–486, Apr. 2019, doi: 10.1007/s11257-018-9217-6.
- [117] P. K. Singh, P. K. D. Pramanik, and P. Choudhury, "An improved similarity calculation method for collaborative filtering- based recommendation, considering neighbor's liking and disliking of categorical attributes of items," *Journal of Information and Optimization Sciences*, vol. 40, no. 2, pp. 397–412, Feb. 2019, doi: 10.1080/02522667.2019.1580881.
- [118] A. Althbiti and X. Ma, "Machine Learning," in *Encyclopedia of Big Data*, L. A. Schintler and C. L. McNeely, Eds. Cham: Springer International Publishing, 2020, pp. 1–5.
- [119] S. Bag, S. K. Kumar, and M. K. Tiwari, "An efficient recommendation generation using relevant Jaccard similarity," *Information Sciences*, vol. 483, pp. 53–64, May 2019, doi: 10.1016/j.ins.2019.01.023.

- [120] J. Feng, X. Fengs, N. Zhang, and J. Peng, "An improved collaborative filtering method based on similarity," *PLoS ONE*, vol. 13, no. 9, p. e0204003, Sep. 2018, doi: 10.1371/journal.pone.0204003.
- [121] B. Kr. Patra, R. Launonen, V. Ollikainen, and S. Nandi, "A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data," *Knowledge-Based Systems*, vol. 82, pp. 163–177, Jul. 2015, doi: 10.1016/j.knosys.2015.03.001.
- [122] R. Farzan and P. Brusilovsky, "Encouraging user participation in a course recommender system: An impact on user behavior," *Computers in Human Behavior*, vol. 27, no. 1, pp. 276–284, Jan. 2011, doi: 10.1016/j.chb.2010.08.005.
- [123] S. Benhamdi, A. Babouri, and R. Chiky, "Personalized recommender system for e-Learning environment," *Educ Inf Technol*, vol. 22, no. 4, pp. 1455–1477, Jul. 2017, doi: 10.1007/s10639-016-9504-y.
- [124] N. Albatayneh, K. Ghauth, and F. Chua, "Utilizing Learners' Negative Ratings in Semantic Content-based Recommender System for e-Learning Forum," *Journal of Educational Technology & Society*, vol. 21, no. 1, pp. 112–125, Jan. 2018.
- [125] S. Dwivedi and V. S. K. Roshni, "Recommender system for big data in education," in *2017 5th National Conference on E-Learning & E-Learning Technologies (ELELTECH)*, Hyderabad, India, Aug. 2017, pp. 1–4, doi: 10.1109/ELELTECH.2017.8074993.
- [126] Z. Gulzar, A. A. Leema, and G. Deepak, "PCRS: Personalized Course Recommender System Based on Hybrid Approach," *Procedia Computer Science*, vol. 125, pp. 518–524, 2018, doi: 10.1016/j.procs.2017.12.067.
- [127] M. Qamhieh, H. Sammaneh, and M. N. Demaidi, "PCRS: Personalized Career-Path Recommender System for Engineering Students," *IEEE Access*, vol. 8, pp. 214039–214049, 2020, doi: 10.1109/ACCESS.2020.3040338.
- [128] H. Drachsler, K. Verbert, O. C. Santos, and N. Manouselis, "Panorama of recommender systems to support learning," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2015, pp. 421–451.
- [129] M. M. El-Bishouty, T.-W. Chang, S. Graf, Kinshuk, and N.-S. Chen, "Smart e-course recommender based on learning styles," *J. Comput. Edu.*, vol. 1, no. 1, pp. 99–111, Mar. 2014.
- [130] M. M. Rahman and N. A. Abdullah, "A personalized group-based recommendation approach for Web search in E-Learning," *IEEE Access*, vol. 6, pp. 34166–34178, 2018.
- [131] R. Bodily and K. Verbert, "Review of research on student-facing learning analytics dashboards and educational recommender systems," *IEEE Trans. Learn. Technol.*, vol. 10, no. 4, pp. 405–418, Oct. 2017.



- [132] A. Dutt, M. A. Ismail, and T. Herawan, "A systematic review on educational data mining," *IEEE Access*, vol. 5, pp. 15991–16005, 2017.
- [133] S. BAher and L. L.M.R.J, "A Comparative Study of Association Rule Algorithms for Course Recommender System in E-learning," *IJCA*, vol. 39, no. 1, pp. 48–52, Feb. 2012, doi: 10.5120/4788-7021.
- [134] A. R. Sharma and P. Kaushik, "Literature survey of statistical, deep and reinforcement learning in natural language processing," in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, Greater Noida, May 2017, pp. 350–354, doi: 10.1109/CCAA.2017.8229841.
- [135] D. Billsus, "User Modeling for Adaptive News Access," *User Modeling and User-Adapted Interaction*, vol. 10, no. 2/3, pp. 147–180, 2000, doi: 10.1023/A:1026501525781.
- [136] S. Shahab, "NEXT LEVEL: A COURSE RECOMMENDER SYSTEM BASED ON CAREER INTERESTS," Master of Science, San Jose State University, San Jose, CA, USA, 2019.
- [137] M. Chen, "Performance Evaluation of Recommender Systems," *IJPE*, 2017, doi: 10.23940/ijpe.17.08.p7.12461256.
- [138] Danah m. Boyd and N. B. Ellison, "Social Network Sites: Definition, History, and Scholarship," *Journal of Computer-Mediated Communication*, vol. 13, no. 1, pp. 210–230, Oct. 2007.
- [139] P. Verduyn, O. Ybarra, M. Résibois, J. Jonides, and E. Kross, "Do Social Network Sites Enhance or Undermine Subjective Well-Being? A Critical Review: Do Social Network Sites Enhance or Undermine Subjective Well-Being?," *Social Issues and Policy Review*, vol. 11, no. 1, pp. 274–302, Jan. 2017, doi: 10.1111/sipr.12033.
- [140] S. G. Mazman and Y. K. Usluel, "Modeling educational usage of Facebook," *Computers & Education*, vol. 55, no. 2, pp. 444–453, Sep. 2010.
- [141] O. Phelan, K. McCarthy, and B. Smyth, "Using twitter to recommend real-time topical news," in *Proceedings of the third ACM conference on Recommender systems - RecSys '09*, New York, New York, USA, 2009, p. 385.
- [142] M. Ghiassi, J. Skinner, and D. Zimbra, "Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network," *Expert Systems with Applications*, vol. 40, no. 16, pp. 6266–6282, Nov. 2013.
- [143] J. Huang, R. Kornfield, G. Szczypka, and S. L. Emery, "A cross-sectional examination of marketing of electronic cigarettes on Twitter," *Tob Control*, vol. 23, no. suppl 3, pp. iii26–iii30, Jul. 2014.

- [144]E. Lahuerta-Otero and R. Cordero-Gutiérrez, “Looking for the perfect tweet. The use of data mining techniques to find influencers on twitter,” *Computers in Human Behavior*, vol. 64, pp. 575–583, Nov. 2016.
- [145]M. Hassan and M. Hamada, “Recommending Learning Peers for Collaborative Learning through Social Network Sites,” in *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, Bangkok, Thailand, 2016, pp. 60–63.
- [146]C. Shen, C.-J. Kuo, and P. T. Minh Ly, “Analysis of Social Media Influencers and Trends on Online and Mobile Learning,” *IRRODL*, vol. 18, no. 1, Feb. 2017.
- [147]L. Qi, Z. Zhou, J. Yu, and Q. Liu, “Data-Sparsity Tolerant Web Service Recommendation Approach Based on Improved Collaborative Filtering,” *IEICE Trans. Inf. & Syst.*, vol. E100.D, no. 9, pp. 2092–2099, 2017, doi: 10.1587/transinf.2016EDP7490.
- [148]Y. Hu, W. Shi, H. Li, and X. Hu, “Mitigating Data Sparsity Using Similarity Reinforcement-Enhanced Collaborative Filtering,” *ACM Trans. Internet Technol.*, vol. 17, no. 3, pp. 1–20, Jul. 2017, doi: 10.1145/3062179.
- [149]H. Al-bashir, M. Abdullateef, A. Romli, and N. Binti, “A Developed Collaborative Filtering Similarity Method to Improve the Accuracy of Recommendations under Data Sparsity,” *ijacsa*, vol. 9, no. 4, 2018, doi: 10.14569/IJACSA.2018.090423.
- [150]S. Ahmadian, M. Afsharchi, and M. Meghdadi, “A novel approach based on multi-view reliability measures to alleviate data sparsity in recommender systems,” *Multimed Tools Appl*, vol. 78, no. 13, pp. 17763–17798, Jul. 2019, doi: 10.1007/s11042-018-7079-x.