Metaheuristics for Collaborative Filtering in Recommender Systems

A thesis submitted during 2021 to the University of Hyderabad in partial fulfillment of the award of a Ph.D. degree in the School of Computer and Information Sciences

by

AYANGLEIMA LAISHRAM

School of Computer and Information Sciences

University of Hyderabad

P.O. Central University, Gachibowli

Hyderabad – 500046

Telangana, India

June-2021

CERTIFICATE

This is to certify that the thesis entitled **"Metaheuristics for Collaborative Filtering in Recommender Systems"** submitted by **AYANGLEIMA LAISHRAM** bearing registration number **14MCPC12** in partial fulfillment of the requirements for award of **Doctor of Philosophy** in the School of **Computer and Information Sciences** is a bonafide work carried out by her under my supervision and guidance.

The thesis is free from plagiarism and has not been submitted previously in part or in full to this or any other University or Institution for award of any degree or diploma. The student has the following publication(s) before submission of the thesis/monograph for adjudication and has produced evidence for the same in the form of acceptance letter or the reprint in the relevant area of her research:

1. Ayangleima Laishram, Satya Prakash Sahu, Vineet Padmanabhan, Siba Kumar Udgata; Collaborative Filtering, Matrix Factorization and Population Based Search: The Nexus Unveilded, ICONIP, 2016, Springer, pp 352-361

2. Ayangleima Laishram, Vineet Padmanabhan, Rajendra Prasad Lal . "Analysis of similarity measures in user-item subgroup based collaborative filtering via genetic algorithm". IJIT, Springer, Dec 2018, Volume 10, Issue 4, pp 523527

3. Ayangleima Laishram, Vineet Padmanabhan. "Discovery of User-Item Subgroups via Genetic Algorithm for Effective Prediction of Ratings in Collaborative Filtering". Applied Intelligence, Springer, 20 May 2019, ISSN 1573-7497

4. Ayangleima Laishram, Vineet Padmanabhan. "Enhanced Collaborative Filtering through User-Item Subgroups, Particle Swarm Optimization and Fuzzy C-Means". CanAI 2019, Springer, pp 94-106.

Further, the student has passed the following courses towards fulfillment of coursework requirement for Ph.D:

Course Code	Name	Credits	Pass/Fail
CS 801	Data Structures and Algorithms	4	Pass
CS 802	Operating Systems and Programming	4	Pass
AI 851	Trends in Soft Computing	4	Pass
AI 852	Learning and Reasoning	4	Pass

Prof. Vineet Padmanabhan
Supervisor

Prof. Chakravarthy Bhagvati
Dean of SCIS

DECLARATION

I, **AYANGLEIMA LAISHRAM**, hereby declare that this thesis entitled **"Metaheuristics for Collaborative Filtering in Recommender Systems"** submitted by me under the guidance and supervision of **Vineet Padmanabhan** is a bonafide research work. I also declare that it has not been submitted previously in part or in full to this University or any other University or Institution for the award of any degree or diploma.

A report on plagiarism statistics from the University Library is enclosed.

Date:

Name: **AYANGLEIMA LAISHRAM**

Signature of the Student:

Reg. No.: **14MCPC12**

Signature of the Supervisor:

To my father, to my mother, who sacrificed many comforts to get a good education for their children.

To my siblings, and my husband. Without their support and encouragement, this would not have been possible.

Abstract

It has become imperative in the current internet based era to advance technology in such a way that the preferences of individuals/users' could be learned from the existing data and recommendations be made on unseen data wherein the user is satisfied with the recommended data/items to a large extend. Recommender systems technology have been put forward by keeping this idea in mind and several multinationals make use of this paradigm to expand their business initiatives. In this thesis we are mainly focused on devising methods that can improve the recommendation as well as prediction accuracy in collaborative filtering (CF) based recommender systems. To achieve this end we propose a variety of algorithms in which metaheuristic techniques are combined with matrix factorisation methods and the combined framework is tested on two main approaches used for collaborative filtering in recommender systems, namely, *model based* and *neighbourhood* based collaborative filtering.

In the case of model based collaborative filtering we demonstrate how metaheuristic techniques like Particle Swarm Optimisation (PSO) and Genetic Algorithm (GA) can be combined with matrix factorisation techniques like Maximum Margin Matrix Factorisation (MMMF). The metaheuristic algorithms such as PSO and GA are exploratory in nature which enhance the traditional model-based collaborative filtering techniques like MMMF with exploitatory nature of gradient descent. The gradient descent approach may get trapped in local optima which is why we plan to employ metaheuristic techniques. Our algorithm starts from multiple initial points and uses gradient information and swarm-search as the search progresses. We show that by this process we get an efficient search scheme to get near

optimal point for maximum margin matrix factorization. Our experimental results on benchmark datasets demonstrate that when the exploration capability of popula- tion based search algorithms is combined with gradient search direction of MMMF, the proposed models are able to achieve better accuracy as can be evidenced from the derived RMSE and MAE values .

We extended the neighbourhood based collaborative filtering technique by adopting the concept of discovering highly correlated user-item subgroups. Our proposal of constructing the user-item-subgroup based collaborative filtering can be done in two ways, namely, through a two-step approach and a fuzzy c-means clustering approach. In the two-step process, we proposed different algorithms to identify the highly correlated user-item subgroups. Then, we used least squares method to predict the missing ratings by using the rating information of the highly correlated user-item subgroups. In fuzzy c-means clustering based user-item subgroup algorithm, the highly correlated user-item subgroups are discovered in one step. We optimized the initialization of centroids in fuzzy c-means by using particle swarm optimization to accurately discover highly correlated user-item subgroups in CF. We observed that our proposed algorithms in the two step approach outperforms all the CF models under comparison for all benchmark datasets. Our findings in terms of MAP suggest that the correlation of the subgroups discovered by GA that evaluates fitness by calculating mean squared residue and row variance is significant though the effectiveness is less for smaller dataset. In the case of fuzzy c-means approach, the metaheuristic optimization algorithm acts as a booster to improve the fuzzy c-means clustering in discovering highly correlated user-item subgroups by initializing the initial centroid of the clusters to the nearest optimal solutions. Our experimental results have shown a promising way of making use of user-item subgroups in helping to capture highly similar user preferences on a subset of items.

Acknowledgements

The accomplishment of this doctoral thesis could not have been possible without a collective support and blessings of various people during my journey of Ph.D. I feel overwhelmed and pleased to acknowledge each and everyone for their kind contribution.

The foremost is my supervisor, Prof. Vineet Padmanabhan, I would like to express my sincere gratitude to him for his invaluable guidance, constant support, and patience. I am extremely grateful that he took me on as his student and continued to have faith in me over the years. I have benefitted greatly from his wealth of knowledge in research and meticulous editing. I wish to acknowledge him for being kind and give me firm guidance throughout my PhD journey and in my career path.

I would like to extend my gratitude to my doctoral committee members, Prof. Siba K. Udgata and Dr. Sobha Rani T., for their valuable suggestions during my entire PhD journey. I extend admiration to my supervisor of M.Tech at International Institute of Information Technology, Bhubaneshwar, Dr. Swati Vipsita for her inspiration to research in the Machine Learning and Soft Computing fields. My gratitude also goes to my senior Dr. Tadiparthi V.R. Himabindu for being available to answer my doubts academically. I would like to honestly praise my colleagues without whom my PhD journey would have been a challenging one in both inside and outside the lab. In particular, Ms. Thoudam Jomita Devi, Ms. Bebina Majhee, Ms. Siltu Laishram, Mrs. Pabitra Sharma, Ms. Shijagurumayum Dharmajyoti Sharma, Mr. Abdul Basit, Mr. Abdulsalam Alammari, Mrs. Laiphangbam Melinda, Mr. Raghu Ghanapuram, Dr. Venkatesh Pandiri, Mrs. Sowmini Devi, Mr. Salman KH, Dr. Vikas Panjey, Dr. Venkat

Kagita, Mr. Vipul Maheshwari, Mr. Harsh Maheswari, Ms. Jaya Bhushan, Mr. Abhishek Shukla, and Mr. Sandeep Dey.

I am extremely thankful to my husband Dr. Debabrot Borgohain who always inspire me to be a better person. My gratitude also extends to my brothers Laishram Chingsanglakpa, Laishram Kiyamba and their wives for morally supporting me. I extend the deepest salutation to my father Dr. Laishram Lokendro and mother Nahakpam Shantibala Devi who constantly encouraged me to pursue higher studies, and also for being there for me at every stage of my life. I wish to express that I am indebted and owe so much to my parents such that my depth of expression acknowledgment will always be small when compared to their contributions of parenting. With immense pride, this doctoral dissertation is dedicated to my parents for providing me the strong roots to standalone and sustain in such a long journey of life.

Ayangleima Laishram

Contents

Declaration **iv**

Abstracts **vi**

Acknowledgements **viii**

List of Figures **xiii**

List of Tables **xiv**

1 Introduction **1**

1.1 Motivation 4

1.2 Thesis Contributions 6

1.3 Structure of the Thesis 9

1.4 Publications of the Thesis 10

2 Fundamental Concepts and Related Work **11**

2.1 Recommender Systems 11

2.2 Collaborative Filtering 13

2.2.1 Model-Based Collaborative Filtering 13

2.2.2 Neighbourhood-based Collaborative Filtering 17

2.2.3 Model-based vs Neighbourhood-based Collaborative Filtering Algorithms 20

2.3 Metaheuristics 21

2.3.1 Particle Swarm Optimization 23

2.3.2 Genetic Algorithm 25

2.3.3 Combining Metaheuristics with Collaborative Filtering 28
2.4 Shortcomings of Collaborative Filtering 29
2.5 Evaluation of Collaborative Filtering Algorithms 29
2.5.1 Datasets 29
2.5.2 Experimental Settings 31
2.5.3 Evaluation Measures Discussion 35
2.6 Summary 39

3 Metaheuristics for Matrix Factorization 41
3.1 Proposed Algorithms 42
3.1.1 Gradient Descent Particle Swarm Optimization based MMMF (GDPSO-MMMF) 44
3.1.2 Particle Swarm Optimization with Mutation based MMMF (PSOm-MMMF) 45
3.1.3 Heirarchical Particle Swarm Optimization based MMMF (HPSO-MMMF) 49
3.1.4 Genetic Algorithm based MMMF 50
3.2 Experimental Settings and Results 52
3.2.1 Parameters Settings 54
3.2.2 Results and Analysis 54
3.3 Summary 57

4 A Two-Step Process of Discovering User-Item Subgroups Using Metaheuristics in Neighbourhood Method 59
4.1 Background: Local Least Squares Imputation of Collaborative Filtering 61
4.1.1 Discover similar users 61
4.1.2 Prediction of missing ratings by using least squares method . 62
4.2 Proposed Algorithms 64
4.2.1 Subgroup based Local Least Squares Imputation of Collaborative Filtering (SLLSimCF) 66
4.2.2 Genetic Algorithm oriented Subgroup based Local Least Squares Imputation of Collaborative Filtering (GA-SLLSimCF) 68
4.2.3 Iterative Subgroup based Local Least Squares Imputation of Collaborative Filtering (ISLLSimCF) 71

4.2.4 Iterative based Genetic Algorithm oriented Subgroup based Local Least Squares Imputation of Collaborative Filtering (I-GA-SLLSimCF) 73
4.3 Empirical Analysis of User-Item Subgroups Algorithms via Two-Step Process for Prediction and Recommendation 75
4.3.1 Experimental Settings 75
4.3.2 Parameter Setting 75
4.3.3 Comparison 79
4.3.4 Results and analysis 80
4.4 Summary 87

5 Discovery of User-Item Subgroups through Co-Clustering and Metaheuristics in Neighbourhood based Collaborative Filtering 89
5.1 Co-Clustering 90
5.2 Proposed Algorithm 93
5.3 Experimental Settings and Results 95
5.3.1 Experimental Setup 95
5.3.2 Experimental Results and Discussions 97
5.4 Summary 103

6 Conclusions and Future Work 104

References 108

List of Figures

3.1 GDPSO-MMMF framework 46
3.2 PSOm-MMMF framework 48
3.3 HPSO-MMMF framework 51
3.4 Heirarchical Tree Structure of neighbourhood topology 52
3.5 GA-MMMF framework 53
3.6 RMSE and MAE for MMMF, GDPSO-MMMF, PSOm-MMMF, HPSO-MMMF and GA-MMMF for 10k dataset 56
3.7 RMSE and MAE for MMMF, GDPSO-MMMF, PSOm-MMMF, HPSO-MMMF and GA-MMMF for 50k dataset 56
3.8 RMSE and MAE for MMMF, GDPSO-MMMF, PSOm-MMMF, HPSO-MMMF and GA-MMMF for 100k dataset 57

4.1 Block diagram of LLSimCF 61
4.2 Block diagram of the proposed method 66
4.3 Graphs show parameter tuning results of several parameters used in the proposed algorithms *GA-SLLSimCF*, *ISLLSimCF* and *SLLSimCF* . . . 78
4.4 Recommendation performance comparison of different CF models based on *Mean Average Precision* (MAP) with respect to different number of subgroups. 86

5.1 Block diagram of the proposed method, PSO-CoC 94
5.2 Performance of the proposed algorithm (*PSO-CoC*) with three well known CF models across different number of subgroups. 98
5.3 Performance of the proposed algorithm (*PSO-CoC*) with different number of subgroups for base CF models. 101

List of Tables

2.1 Statistics related to the datasets being used in this thesis 30

3.1 Comparison of the proposed algorithms with MMMF and [19] in terms of RMSE for different dataset sizes. 57
3.2 Comparison of the proposed algorithms with MMMF and [19] in terms of RMSE for different dataset sizes. 57

4.1 *RMSE* of the proposed algorithms for the benchmark dataset, *Movielens*. 82
4.2 *MAE* of the proposed algorithms for the benchmark dataset, *Movielens*. 82
4.3 Results related to different similarity measures of the proposed algorithm on benchmark dataset, *Movielens*. 83
4.4 Comparison of the proposed algorithm with base LLSimCF and item-based CF [55] on benchmark dataset, *Movielens*. 83
4.5 Comparison of the proposed algorithm in terms of recommendation accuracy 83

5.1 Comparison of the algorithms *MCoC*, *MCoC-UU-II-UI* and *SCoC* and the proposed algorithm (*PSO-CoC*) with base algorithm for predicting missing values as *MMMF* 101
5.2 Comparison of the algorithms *MCoC*, *MCoC-UU-II-UI* and *SCoC* and the proposed algorithm (*PSO-CoC*) with base algorithm for predicting missing values as *PMF* 102
5.3 Comparison of the algorithms *MCoC*, *MCoC-UU-II-UI* and *SCoC* and the proposed algorithm (*PSO-CoC*) with base algorithm for predicting missing values as *IB* 102

Chapter 1

Introduction

These days it is quite evident that the world wide web plays a significant role in our everyday life. It can be said indisputably that a number of popular applications existing today, which we use on a daily basis, for web search are enhanced by the *Recommender Systems (RS)* technology. Many of these applications have to deal with massive amounts of data and this leads to the *information overload* problem wherein the *target* information gets buried in an enormous amount of *irrelevant* information. The task of a recommender system is to suggest *possibly interested* unseen `items` to `users` and help them in their decision making process. For instance, in a movie recommender system, one way of doing the recommendation would be that if two users have both liked certain common movies, then the movies that the first user has liked but the second one has not yet seen/rated can be recommended to the second person. This type of recommendation is *collaborative* in nature wherein similarities between users/items are taken into consideration for making recommendations. Machine learning techniques have been heavily used by researchers in solving the challenges involved in building *collaborative filtering* based Recommender Systems.

In general, there are two strategies for developing Recommender System algorithms, namely *Collaboraive Filtering (CF)* [46] and *Content-Based (CB)* [49]. The CB approach basically builds profiles for users or items based on the explicit information provided and perform profile matching to make recommendations. The CB approach needs external information about the users or items that might be unavailable. On the other hand, the CF approach, as mentioned above relies only on the user-item interactions to make recommendations [43].

Collaborative filtering algorithms can further be subdivided into 1) *model-based* and 2) *neighbourhood-based* algorithms [11]. In the model-based approach, machine learning algorithms are used to first develop a model of user ratings so as to predict the user's rating of unrated items. *Matrix factorization* methods are the most successful techniques developed for model-based collaborative filtering algorithms in which a factor model is built so as to fit the data for prediction of missing ratings. To get such an accurate model, the objective function of the model has to be minimized to reduce the errors of the predictions of the missing ratings. In the neighbourhood-based approaches, similarities between users or between items are measured for further use of the information in the prediction or for computing the recommendation. The most widely applied techniques in neighbourhood based approach are *user-based* and *item-based* approaches. In a typical neighbourhood based approach, an assumption is made that all the items that are liked by a group of users' who are similar to the *active user* (user to whom recommendation is to be made) will be recommended to the active user also. This assumption is sometimes not tenable [12, 68] and therefore researchers aim at finding highly correlated subgroup of users and items to develop recommender systems that helps in improving the accuracy in the recommendations. Biclustering technique has been widely used to identify such correlations among users and items as can be seen from the literature [5, 17, 36, 59].

The main aim of the current study is to explore how metaheuristic techniques like Particle Swarm Optimisation (PSO) and Genetic Algorithm (GA) can be combined with collaborative filtering methods to address accuracy issues in recommender systems. For instance, from the literature it can be seen that, particle swarm optimisation technique has been used in recommender systems that are *usage-based* [4][22] as well as in recommender systems that learn personal opinions to make specific recommendations [64]. Similarly, PSO techniques have also been used to enhance scalability issues in recommender systems as can be evidenced from [6, 66]. In [20] matrix factorisation based collaborative filtering model is combined with PSO wherein Singular Value Decomposition (SVD) is used for factorisation of the matrix. The SVD kind of factorisation approach cannot be equated with the recent matrix factorisation techniques like maximum margin matrix factorisation (MMMF) being deployed in collaborative filtering because SVD do not address the issues wherein the solution space has multiple local minima [60]. It should be kept in mind that most of the benchmark

datasets that are currently being used to validate recommender system algorithms are usually sparse. Therefore it is worth exploring the benefits of combining metaheuristic techniques with matrix factorisation methods like MMMF.

There is also evidence for combining evolutionary methods like genetic algorithm with collaborative filtering as proposed in [47]. What we have noticed is that though few works as outlined above have addressed the problem of combining metaheuritics and collaborative filtering in model-based strategies this is not the same with neighbourhood based strategies. We found very little research that analyzes the neighbourhood method in capturing the duality between users and items with the help of metaheuristic techniques like PSO/GA. Although, as mentioned earlier, [17] presented an immune inspired algorithm of biclustering in collaborative filtering and [57] employed a modified cuckoo search technique by combining with fuzzy c-means in recommender systems, combining metaheuristic techniques with different clustering strategies has not been explored yet. We aim to fill this gap by exploiting metaheuristic techniques in finding partial matching preferences of users in a collaborative filtering model so as to improve the accuracy in the recommendations.

The main role of a recommender system is in either performing *prediction* or *recommendation*. In the *prediction* task, the algorithm estimates the possible ratings that the user may give to an item. In the *recommendation* task, the algorithm predicts ratings for all the unseen items and recommends a list of top rated items from the group of unseen items assuming that the user will like it. To develop a good collaborative filtering algorithm that can solve various recommendation/prediction tasks, one needs to know the type of *feedback* he/she is dealing with. The type of feedback can effect the *performance* of an algorithm. An algorithm that works well with a dataset having one type of feedback may not work well with another algorithm developed to perform a particular task but has a different type of feedback. Therefore choosing the right type of feedback for a specific task is preferred in evaluating the performance of recommendation algorithms. There are two types of feedback in recommender systems, namely *explicit feedback* and *implicit feedback*. In explicit feedback the user gives feedbacks intentionally for items. It is further classified into two types such as numerical feedback and binary feedback. In numerical feedback the user expresses feedback in the form of ratings whereas in binary feedback the user gives information based on his/her likes/dislikes of items. Implicit feedback is derived based on user actions like browsing

history, purchase history, wish-list etc. on the system [43]. In this thesis we make use of the explicit feedback in the form of numerical ratings which helps in improving the prediction accuracy of the recommender system.

The evaluation measures used for evaluating the performance of recommender system algorithms are borrowed from statistics and information retrieval. There are several measures to evaluate recommender systems from different aspects such as *coverage, accuracy, novelty, serendipity, diversity, confidence* etc. as proposed in the literature [58]. Among the measures mentioned, the most widely used measure is the accuracy measure despite the possibility of evaluation from different perspectives. In prediction computation, *accuracy* measures such as *Root Mean Squared Error (RMSE)* and *Mean Absolute Error (MAE)* are the most widely employed metrics whereas in recommendation task metrics like *precision*, *recall* and *F1-Score* are the most popular ones. The key points involved in developing and evaluating recommender system algorithms are choosing the right accuracy measure for evaluation based on a particular task and right type of feedback in datasets. In this thesis we deal with both recommendation and preiction task and therefore the accuracy measures used for the prediction task are *RMSE* and *MAE* whereas for recommendation *precision*, *recall*, and *F1-Score* are used.

1.1 Motivation

As explained above, collaborative filtering algorithms are the most popular recommendation algorithms as it can make decently accurate predictions by only finding and analyzing users' past behaviour. The key points that motivate us to employ metaheuristic techniques in both model-based and neighbourhood based collaborative filtering algorithms can be outlined as follows:

- Users give ratings to items in the form of explicit feedback in collaborative filtering based recommender systems. The ratings are given based on a scale of how much the user likes the item and this makes the algorithm easier to judge the preferences of users. For example, in a rating scale of [1-5] the rating 1 represents the least preferred item and the rating 5 represents the most preferred item. However in reality, the number of items that receive ratings are less leading to

the *data sparsity* problem and therefore it is challenging to build a good recommendation algorithm that gives accurate result by working on a sparse dataset. In recommender systems, only partial truth about the users and items are available and the recommendation algorithms need to work with this imprecise data and partial information to find an optimal solution. Hence we need to explore appropriate techniques that are tolerant to imprecise data and develop algorithms that can provide near optimal solutions to solve such problems. Metaheuristic techniques can help us in this endeavour.

- Metaheuristic optimization algorithms are known for successfully solving real time complex problems with partial truth efficiently as compared to traditional optimization techniques. Metaheuristic algorithms start by randomly initializing a set of candidate solutions called *population*, on a high dimensional search space instead of starting from a single point as in the case of traditional optimization techniques. Metaheuristic algorithms take advantage of the dynamics of population of candidate solutions that cooperate with each other to find better solutions. In addition to that, metaheuristic algorithms perform well even when only partial information about the problem is known, and come up with a near-optimal solution in which the traditional optimization techniques fail to do so without fulfilling the standard conditions. Thus, it is a good idea to leverage the capability of the metaheuristic algorithm in a problem domain like recommender system wherein based on only the partial information about the users there is a need to find approximate ratings for the unseen items.

- In model-based collaborative filtering, the accurate prediction in recommender system is obtained by minimizing the loss function of fitting a target matrix to an approximate matrix. We employ matrix factorization, as it is one of the most successful realizations of model based collaborative filtering. We focus on finding a suitable way to combine metaheuristic algorithms with search direction of factor matrices to improve the performance.

- Typically, neighbourhood-based collaborative filtering algorithms assume that the entire preferences of users that are similar will be matching with those of an active user. However, this assumption is not always reasonable and possibly

may lead to less accurate predictions due to the inclusion of irrelevant information in the prediction computation. Therefore, we focus on identifying highly correlated group of users based on a subset of items for an *active user* to predict the missing ratings accurately. We propose user-item subgroup based framework and compare with traditional collaborative filtering algorithms to judge the performance.

1.2 Thesis Contributions

This dissertation aims in exploring the performance of collaborative filtering algorithms when combined with metaheuristic optimization techniques for recommendation/prediction in recommender systems. It also aims in discovering highly correlated user-item subgroups to find closer relationships among the Users/Items so as to improve the accuracy in recommender systems. We compare the proposed metaheuristic based frameworks and user-item subgroup based algorithms with the standard (benchmark) algorithms to judge the performance of the proposed framework. We have summarized the contributions below:

1. *Metaheuristics for Matrix Factorization*
   We combine different variants of metaheuristic optimization techniques with matrix factorization to observe the corresponding improvements in the prediction of missing ratings. Metaheuristic algorithms are problem independent algorithms wherein the objective function is derived according to the problem at hand. Therefore metaheuristic techniques can be applied to any collaborative filtering algorithms for enhancing the search process so as to find the optimal solution. We employ popular metaheuristic algorithms such as PSO and GA, and propose four collaborative models by combining the population based search algorithms with the most successful matrix factorization technique, namely Maximum Margin Matrix Factorization (MMMF). In particular, we propose three variations of PSO based MMMF, each varying in social structure of particles that results in increasing swarm diversity. We have applied such population based search algorithms to find low rank latent factors for prediction of missing ratings in a

user-item rating matrix. The different variants of the metaheuristic algorithms that we adopt in order to find the latent factors are summarized below.

- Local Best Structure of PSO [7]: In this variant, initial neighbourhoods of particles in the swarm are created randomly. The particle with the best fitness value in a neighbourhood is selected as the local best particle, and navigation is done towards the rest of the particles in the neighbourhood. All local best particles share their information and the one with the best fitness value is selected as the global best particle. Entire population is updated with the best particle's position in each step and therefore there is an influence to move towards the global best solution effectively.
- PSO with Mutation Operator [7, 48]: To avoid getting trapped in the local optima due to the loss of diversity in the swarm, an operator called *mutation* operator from evolutionary algorithms is adopted in the base PSO method. In this variant of PSO, particles are randomly initialized based on the unsatisfactory criteria of the swarm threshold so as to preserve diversity in the swarm.
- Hierarchical Neighbourhood based PSO [23]: The social structure of the particles is constructed based on a hierarchical tree structure wherein a neighbourhood is formed by a parent with immediate children such that the parent becomes the local best particle in the neighbourhood and the root of the tree represents the global best particle.
- Genetic Algorithhm [47]: Evolutionary algorithm is used to find the optimal latent factors through the powerful searching imitation of genetic evolution wherein the principle followed is that of survival of the fittest.

The main objective of the proposed framework is to find whether there is any significant improvement in the performance of the collaborative filtering algorithms when combined with heuristic search in a model-based framework. The proposed algorithms and experimental analysis related to metaheuristic based MMMF are discussed in Chapter 3.

2. *A Two-Step Process of Discovering User-Item Subgroups Using Metaheuristics in Neighbourhood based Collaborative Filtering.*

In the neighbourhood-based approach, traditional collaborative filtering considers the entire ratings of similar users or items in the prediction of missing ratings. This seems not reasonable because if the preferences of two users are only matching on some items then that does not necessarily mean that their preferences will be matched on the entire set of items. Therefore, for prediction of missing ratings, we propose collaborative filtering based frameworks that consider information only about a subset of similar users based on a subset of items. A subset of similar users based on a subset of items are termed as *user-item subgroup*. The proposed algorithm basically takes two steps to discover correlated user-item subgroups. First, it discovers a group of similar users based on the entire consumed items of the active user. Then, it selects only a subset of similar items based on the selected subset of users. Finally, the rating information of the subgroup is used to predict the missing ratings by using *least squares* approach.

The proposed algorithms are based on several techniques such as (1) using a threshold to find the highly correlated subset of items among selected k similar users, (2) having an iterative approach to enhance the selection of k subsets of similar users, (3) utilizing metaheuristic algorithms like Genetic Algorithm to select the correlated subset of items, and (4) a hybrid approach of the iterative technique and the Genetic Algorithm. Popular similarity measures like Pearson correlation, Adjusted cosine and Euclidean distance measures are used with the proposed algorithms and a performance comparison is done with the existing subgroup based recommendation algorithms as well as the traditional collaborative filtering algorithms. The proposed algorithms are evaluated for both the prediction and recommendation accuracy to show robustness of the proposed method. The details of these algorithms and the evaluation results are discussed in Chapter 4.

3. *Discovery of User-Item Subgroups through Co-Clustering and Metaheuristic in Neighbourhood based Collaborative Filtering.*
   As mentioned above, in Chapter 4, we discovered highly correlated user-item subgroups through a two step process in neighbourhood based collaborative filtering algorithms. To avoid the two-step process, we employ Fuzzy C-Means (FCM) clustering which is further enhanced by Particle Swarm Optimization to

discover correlated user-item subgroups in one step though the method is executed through several iterations in order to refine the selected subgroups. The metaheuristic algorithm, PSO, is applied to obtain optimal centroid positions for FCM clustering. We initialize the initial centroids of the FCM clustering to the optimal positions found by PSO to achieve an effective clustering method in finding highly correlated user-item subgroups. Once the correlated subgroups are found, the missing ratings in the subgroups are predicted by using rating information of the subgroups rather than including irrelevant information from all the consumed items of the active user as in the case of traditional collaborative filtering methods. The proposed algorithm is compared with existing subgroup based collaborative filtering algorithms and traditional collaborative filtering algorithms . The proposed algorithms are evaluated for recommendation accuracy and the details are discussed in Chapter 5.

1.3 Structure of the Thesis

The thesis is organized in the following way: Apart from Chapter 1 which is introductory in nature, Chapter 2 discusses the basic information related to recommender systems, the purpose for which such systems are built as well as an explanation related to some of the existing recommendation algorithms. It also discusses metaheuristic optimization algorithms and some of the recent metaheuristic based recommendation algorithms. Different similarity measures and evaluation measures for collaborative filtering are also discussed. Chapter 3 presents our proposed work on model-based collaborative filtering, namely matrix factorization, combined with several variants of metaheuristic algorithms such as gradient descent based particle swarm optimisation, mutation based particle swarm optimisation, hierarchical particle swarm optimisation and genetic algorithm. The experimental analysis of the proposed algorithms have also been discussed. In Chapter 4, we present recommendation algorithms that focus on identifying partial matching of the preferences among the users and the items in a two-step process by using various techniques including metaheuristic algorithms like Genetic Algorithms. The identified information regarding partial matching of preferences is further used in the computation of prediction and recommendation of items. In Chapter 5, we propose a method for identifying such partial matching of preferences

through co-clustering. The proposed method makes use of co-clustering in the first step and thereafter combines it with metaheuristic techniques such as PSO to further analyze the performance. We conclude with Chapter 6 wherein a summary of the research contributions related to the thesis is outlined as well as pointers for future work is provided.

1.4 Publications of the Thesis

1. Ayangleima Laishram, Satya Prakash Sahu, Vineet Padmanabhan, Siba Kumar Udgata. "Collaborative Filtering, Matrix Factorization and Population Based Search: The Nexus Unveiled". International Conference on Neural Information Processing, ICONIP 2016, Springer, pp 352-361.

2. Ayangleima Laishram, Vineet Padmanabhan, Rajendra Prasad Lal . "Analysis of similarity measures in user-item subgroup based collaborative filtering via genetic algorithm". International Journal of Information Technology, Springer, December 2018, Volume 10, Issue 4, pp 523527

3. Ayangleima Laishram, Vineet Padmanabhan. "Discovery of User-Item Subgroups via Genetic Algorithm for Effective Prediction of Ratings in Collaborative Filtering". Applied Intelligence, Springer, 20 May 2019, ISSN 1573-7497

4. Ayangleima Laishram, Vineet Padmanabhan. "Enhanced Collaborative Filtering through User-Item Subgroups, Particle Swarm Optimization and Fuzzy C-Means". Canadian Conference on Artificial Intelligence 2019, Springer, pp 94-106.

Chapter 2

Fundamental Concepts and Related Work

In this chapter we introduce the fundamental concepts of recommender systems and also outline some existing research that tries to address the recommendation problem from various directions. We initially explain the most widely used strategy in recommender system algorithms, namely collaborative filtering wherein both model-based and neighbourhood-based collaborative filtering are discussed at length. Thereafter, we outline two metaheuristic approaches (Particle Swarm Optimisation and Genetic Algorithms) that can be combined with collaborative filtering and point out some of the earlier works in this direction. We also talk about the various datasets being used in our experiments and describe in detail the evaluation measures suitable for prediction and recommendation in recommender systems.

2.1 Recommender Systems

Recommender Systems (RS) are software tools that deal with massive amount of data to filter irrelevant information and suggest only items that are matching with users' opinions. RS are widely used in several website-based applications to make the users utilize the applications efficiently. Well known companies likes Twitter, Netflix, Google, Amazon etc. are able to improve their performance so as to attract more consumers by enhancing their technology with that of the recommender systems technology. The objective of recommender systems is to provide suggestions of items that are most

likely relevant to the users, thereby easing the decision making process of the users. With the help of recommender systems technology, it becomes easier for the user to find interesting items from the available pile which in turn will also help him/her in saving time as well as tackle the problem of *information overload*. Information overload occurs when a user is fed with massive amounts of data and he/she is not able to make a decision as to which item to buy/consume. Recommender systems technology helps to address the problem of information overload. For example, let's consider a movie web-based scenario such that a user is recommended an unseen movie called `Captain Marvel`, given that the user watched and liked the prequel of the movie called `Iron Man`. It is assumed that the user will like the recommended movie which is based on his/her past preferences. The successful recommendation of the movie/item is due to the accurate prediction of the underlying algorithm that the recommender system makes use of.

Two common filtering techniques that the recommender systems technology use to accomplish such accurate recommendations are *Content-Based (CB)* filtering and *Collaborative Filtering (CF)*. The CB strategy operates by explicitly gathering information/content about users and items to build profiles, and perform a profile matching to make suggestions related to the most interesting items for the users. For example, in a scenario wherein we want to recommend books to a user who has read a particular book **A**, then all the books related to that particular author/genre/publisher etc. can be recommended by using a content based approach. In the case of collaborative filtering, items are recommended to a new user by collecting preferences information from many other users (collaborating) over the items as in the case of books wherein all books read by people who have read book **A** will be considered and the one with the highest count on the top of the list will be recommended. The content-based strategy tends to fail if the information about the user or the item is not sufficient to define the preferences of the users whereas the collaborative filtering strategy depends only on the relationships between users and items to generate unseen interesting items. The basic difference between CB and CF strategies is the way they obtain information about the users/items to make recommendation wherein CB does so by considering matching features in the user and item profiles. CF does in two ways: one by finding a group of similar users or items, and the other by characterising user and item vectors in latent feature space. In this thesis, we focus on collaborative filtering for finding unseen interesting items

for recommendation to users [46]. The collaborative filtering approach is discussed at length in the following sections.

2.2 Collaborative Filtering

The collaborative filtering approach depends on past user behaviours and also analyzes the behaviour of other users with an aim to make recommendations. As mentioned in Section 2.1, CF can be categorized into two approaches, namely *model-based* and *neighbourhood-based*. The model-based strategy learns hidden features of past activities of users in order to build a model for recommendation. In neighbourhood-based strategy, the relationships between users and items are analyzed to find a group of similar users or alternatively items. The two strategies are thoroughly analyzed in the following sections.

2.2.1 Model-Based Collaborative Filtering

The model-based collaborative filtering strategy applies machine learning techniques to build a model by using past information of users and items where the model is learnt and thereby missing ratings are accurately predicted. The model can be developed by applying a variety of machine learning techniques like 1) *matrix decomposition* such as SVD [54], PCA [26], 2) *Latent Semantic techniques* [30], 3) *Bayesian Networks* [11], 4) *Matrix Factorization* [43, 45, 52, 53] and 5) *Clustering* techniques [56, 65, 68, 69]. Of these, the *Matrix Factorization (MF)* method is one of the most popular model-based CF strategies [67]. The framework of MF allows to have an infinite number of hidden features of the users and items that indicate their characteristics, while only a small number of features can actually influence the preferences to make suggestions. In MF, the users and items are characterized in a hyper-dimensional latent space by using generated hidden features, and the items in the proximity of a user are recommended. Loss functions such as *sum-squared error, hinge loss* etc. are employed to learn the hidden features of the users and items to predict the missing ratings accurately. The constraint of MF on norm of feature vectors leads to convex optimization problems which is efficient [60]. Thus, we focus on matrix factorisation techniques and further improve the framework by combining it with metaheuristics.

2.2.1.1 Matrix Factorisation

In collaborative filtering, it is often the case that most of the entries in the rating matrix are unobserved and this leads to non-convex optimization problems even if we try to minimise the error using a loss function like *squared error*. When compared to other factorisation techniques like Singular Value Decompositon (SVD), matrix factorisation is superior as MF yields convex optimization problems by constraining the norms of the factor matrices[60]. In matrix factorisation, the ratings of users and items are represented in a matrix $Y \in \mathbb{R}^{n \times m}$ such that the number of users and items correspond to n rows and m columns respectively and ratings correspond to the entry values. The entry y_{ij} range from $\{0, 1, 2, \ldots, R\}$ which define the preference degree of i^{th} user on j^{th} item whereas the rating 0 indicates the unknown ratings of the i^{th} user on j^{th} item. Given a partially filled user-item rating matrix Y, the objective of matrix factorisation is to determine the two matrices $U \in \mathbb{R}^{n \times f}$ and $V \in \mathbb{R}^{m \times f}$ such that the dot product of U and V is approximately equal to Y, i.e.

$$Y \approx UV^T \tag{2.1}$$

where inner dimension f is called *numerical rank* of the matrix such that $f < [n, m]$ and hence matrix factorization is inexpensive. The factor matrices U and V are unknown and need to be predicted. The dot product of U and V, $X = UV^T$, is called low-rank approximation of rating matrix Y. The U_i and V_j represent the latent feature vectors of i^{th} user and j^{th} item respectively. In matrix factorisation, user i $(1 \leq i < n)$ and item j $(1 \leq j < m)$ are represented by $u_i \in \mathbb{R}^f$, the i^{th} column of the matrix U^T, and $v_j \in \mathbb{R}^f$, the j^{th} column of the matrix V^T respectively. For example, if we consider the matirx to represent a movie dataset, then each dimension of the vector v^j represents a feature of movie j, such as *genre* namely *romance, comedy, thriller* etc. The preferences of users about the features of movies are indicated by the corresponding entries in the user vector u_i. Thus, the final rating of the user u_i on a movie v_j is the linear combination of the feature vectors such that

$$x_{ij} = \sum_{l=1}^{f} u_{il} v_{jl} = u_i v_j^T \tag{2.2}$$

The matrix factorisation approach uses different objective functions to approximate the predicted rating so that it is close to the true rating and also induce different constraints on the factor matrices U and V to achieve this approximation. Based on the kind of objective functions being used and constraints imposed, some variants of matrix factorisation as given in the literature can be outlined as follows:

1. *Regularized Matrix Factorization (RMF)*

   This method is the simplest among all the matrix factorisation approaches. To find the factor matrices U and V as given in Eqn. (2.1), the sum squared error loss function is used and the resulting function can be described as in Eqn (2.3) [67]:

   $$\mathcal{J} = \min_{U \in \mathbb{R}^{n \times f}, V \in \mathbb{R}^{m \times f}} \frac{\lambda}{2}(||U||_F^2 + ||V||_F^2) + \sum_{ij \in S}(y_{ij} - u_i v_j^T)^2 \tag{2.3}$$

   where λ denotes regularization parameter, S denotes the observed entries at i^{th} row and j^{th} column and $||.||_F$ represents the Frobenius norm where $||U||_F = \sum_{pq} u_{pq}^2$. The ***RMF*** model fits the predicted ratings to the true ratings such that the dot product, $u_i v_j^T$, of the user u_i and item v_j, given in Eqn. (2.1), should be as close as possible to the true rating y_{ij}. The Frobenius norm is used as a regularization term to constraint the norms of U and V so that overfitting can be avoided and this in turn yields better generalization performance.

   The RMF uses gradient descent technique to minimize the objective function. The latent factor matrices U and V are randomly initialized and these matrices are learned iteratively to minimize the objective function. The new U matrix is found by using Eqn (2.4). Similar computation is carried out to find the new V.

   $$u_{il}^{(t+1)} = u_{il}^{(t)} - \tau \frac{\partial L}{\partial u_{il}^t} \tag{2.4}$$

2. *Maximum Margin Matrix Factorization (MMMF)*

   Maximum Margin Matrix Factorisation was first proposed by Nathan Srebro et. al. [60] as a matrix factorisation technique for collaborative filtering. The MMMF model tries to fit the predicted matrix X to the true rating matrix Y by minimizing the hinge loss function as an objective function that consists of a

collection of Support Vector Machines (SVMs). The MMMF relaxes the dimensionality of the feature vectors U and V but constraints the norms while predicting with maximum margin of hyperplanes. In this thesis, we adopt maximum margin matrix factorisation method, since the objective function of MMMF is more appropriate for non-binary ratings as compared to other variants of matrix factorisation methods like RMF and NMF. In MMMF, an additional matrix called a threshold matrix Θ is used in addition to the user U and item V matrices. The threshold matrix has a set of thresholds that are defined in the range $\Theta_{ib}(1 \leq b \leq R-1)$ and the thresholds are also learned to predict a discrete rating $[1, \ldots, R]$. The objective function of MMMF is to minimize the function $\mathcal{J}$ with respect to U,V, and Θ as shown in Eqn (2.5).

$$\mathcal{J}(U, V, \Theta) = \lambda(||U||_F^2 + ||V||_F^2) + \sum_{b=1}^{R-1} \sum_{ij \in S} h(T_{ij}^b[\Theta_{ib} - u_i v_j^T]) \tag{2.5}$$

where the regularization term is the Frobenius norm $||.||_F$ and λ represents regularization parameter, S represents the observed entries such that $ij|y_{ij} > 0$. The smoothed hinge loss function $h(.)$ and T_{ij}^b are defined by Eqn (2.6).

$$T_{ij}^b = \begin{cases} +1, if & b \geq y_{ij} \\ -1, if & b < y_{ij} \end{cases} \quad , \quad h(z) = \begin{cases} \frac{1}{2}(1-z), & if \quad z<0 \\ 0, & if \quad z>1 \\ \frac{1}{2}(1-z)^2, & otherwise \end{cases} \tag{2.6}$$

The factor matrices U, V and threshold matrix Θ are learned iteratively with gradient descent as shown in Eqn (2.7)

$$U_{t+1} = U_t - c\frac{\delta \mathcal{J}}{\delta U}, \quad V_{t+1} = V_t - c\frac{\delta \mathcal{J}}{\delta V}, \quad \Theta_{t+1} = \Theta_t - c\frac{\delta \mathcal{J}}{\delta \Theta} \tag{2.7}$$

From the loss function given in Eqn. (2.5), it can be observed that unlike RMF that focuses more on the specific values of $u_i v_j^T$, MMMF does not need the predicted value $u_i v_j^T$ to be close to the true value y_{ij}. Instead, MMMF compares the predicted rating $u_i v_j^T$ with the threshold matrix Θ_{ib} which is equivalent to a hyperplane in SVM framework such that $u_i v_j^T$ should be as large as possible when comparing with the threshold b and when the condition $b \leq y_{ij}$ satisfies. Similarly $u_i v_j^T$ should be as small as possible for $b \geq y_{ij}$.

3. *Non-negative Matrix Factorization (NMF)*

   Nonnegative Matrix Factorisation also uses sum squared error as loss function wherein it constrains the values contained in the factor matrices U and V to be non-negative [45]. Moreover, NMF requires the non-negative U and V matrices to satisfy Eqn (2.1). The loss function, given by Eqn (2.8), of NMF should be minimized with respect to *U* and *V*, subject to the constraints $u_{il}, v_{jl} >= 0$, such that $1 \geq i \geq n$, $1 \geq l \geq f$ and $1 \geq j \geq m$. The factor matrices U and V are learned by using the update rules iteratively as given by Eqn (2.9) and Eqn (2.10).

$$\mathcal{J} = ||Y - UV||^2 \tag{2.8}$$

$$u_{il} \leftarrow u_{il} \frac{\sum_j v_{jl} y_{ij} / (u_i^T v_j)}{\sum_b v_{bl}} \tag{2.9}$$

$$v_{jl} \leftarrow v_{jl} \frac{\sum_i u_{il} y_{ij} / (u_i^T v_j)}{\sum_b u_{bl}} \tag{2.10}$$

   Note that both the RMF and NMF models are suitable for binary ratings while the MMMF model is suitable for non-binary ratings.

2.2.2 Neighbourhood-based Collaborative Filtering

In neighbourhood-based collaborative filtering methods, past behaviour between users and items are analyzed to find relationships between users or alternatively between items for computing prediction [11]. To built collaborative filtering techniques for recommender systems using neighbourhood method, there are two well known approaches, namely *user-based (UB)* [8, 11, 29, 34, 69] and *item-based (IB)* [18, 46, 55] methods. Basically, in neighbourhood-based methods, the two steps involved in addressing the recommendation problem can be given as follows:

1. Similarity Computation
2. Prediction or Recommendation Computation

In similarity computation, the closeness of preferences between users or the close characteristic between items are evaluated by using different similarity measures to

find a group of similar users or items. The utilization of the information in similarity computation is different in UB and IB approaches. UB approach utilizes the interested items of similar users in prediction or recommendation as long as the items are unseen with respect to the active user. In the IB approach, unseen items from a similar group of items are recommended to users who liked the items in the group. As mentioned earlier, the main motivation of neighbourhood method is the assumption that users with similar opinions might have similar tastes on unseen items.

There are several ways to evaluate similarities between users or items in neighbourhood method [55] . Here, we discuss three such popular similarity measures in collaborative filtering with respect to user-based approach.

1. *Pearson Correlation*

   In this measure, similarity between two users i and j is evaluated through correlation computation. To do so, mutually rated items by the two users i and j are isolated first and the mutually rated set of items is denoted as I. The correlation similarity between the two users i and j is given by Eqn (2.11).

$$sim(i,j) = \frac{\sum_{it \in I}(R_{i,it} - \overline{R}_i)(R_{j,it} - \overline{R}_j)}{\sqrt{\sum_{it \in I}((R_{i,it} - \overline{R}_i))^2}\sqrt{\sum_{it \in I}((R_{j,it} - \overline{R}_j))^2}} \tag{2.11}$$

   where $\overline{R}_i$ represents the average rating given by the i^{th} user, $R_{i,it}$ represents the rating of user i on item it.

2. *Adjusted Cosine*

   In adjusted cosine similarity measure, two users are depicted as two vectors in n dimensional space wherein the cosine angle between the two vectors represent the similarity degree. The mutually rated items by two users i and j are separated and is denoted by set I. To avoid huge differences in the rating scale given by multiple users on a single item, all the ratings given to an item ($it \in I$) by all the users are averaged, and then the averaged value is subtracted from the actual rating given by a user i to an item it. The similarity between the two users is computed by Eqn (2.12).

$$sim(i,j) = \frac{\sum_{it \in I}(R_{i,it} - \overline{R}_{it})(R_{j,it} - \overline{R}_{it})}{\sqrt{\sum_{it \in I}((R_{i,it} - \overline{R}_{it}))^2}\sqrt{\sum_{it \in I}((R_{j,it} - \overline{R}_{it}))^2}} \tag{2.12}$$

where $\overline{R}_{it}$ represents the average rating of the item it.

3. *Euclidean Distance*

   The distance between two users i and j in Euclidean space represents the similarity degree. The items which are co-rated by the two users i and j are isolated first to compute the similarity degree, which is given by Eqn (2.13).

$$sim(i,j) = \sqrt{\sum_{it \in I} (R_{i,it} - R_{j,it})^2} \tag{2.13}$$

   where $R_{i,it}$ represents the rating given by user i on item it, and I represents the set of co-rated items by users i and j.

In the literature on collaborative filtering methods for recommender systems, there is very little work that has been done in explaining the duality that could exist between users and items wherein finding such relationships could actually improve the quality of a recommender system [17, 36, 61]. This area of research is termed as finding user-item subgroups in collaborative filtering [68] and can be explained as follows.

2.2.2.1 User-Item Subgroup based Collaborative Filtering:

It is to be noted that UB and IB approaches find relationships either among users or among items only, despite the fact that the joint analysis of user and item relationships cannot be ignored as shown in works like [17]. Moreover, both UB and IB approaches fail to capture partial matching of preferences between users and items. It can be seen that traditional collaborative filtering approaches assume that the preferences of a group of similar users would be the same over the entire items with respect to the active user. However, this assumption is not always tenable [68] since a group of users having a strong association based on a subset of items may not imply that they will have strong correlation on the entire set of items. Additionally, it is ideal that a user has a concentrative yet diverse set of preferences rather than a diverse preference over the entire items. Thus, there is a possibility of overlooking a strongly correlated group of users while searching for a group of similar users over the entire items and this could degrade the prediction accuracy in collaborative filtering. Hence, it is reasonable to focus on filtering irrelevant subset of items based on a group of similar users

unlike traditional CF, and utilize the information of discovered subgroup of strongly correlated users and items in further computation.

From the literature it can be found that researchers have used biclustering techniques to capture the partial matching of the uers' preferences. Symeonidis et. al. [62] applied a well known biclustering algorithm called Bimax for constant values to find duality between users and items, and proposed a nearest-bicluster algorithm that captures partial matching of preferences to solve issues related to collaborative filtering. Then Symeonidis et. al. [61] extended the work and applied xMotif biclustering technique for coherent values to capture partial matching of users' preferences. However, the algorithm was restricted to identify overlapping between the biclusters which results in limitation of capturing accurate partial matching of preferences. Pablo A.D. et. al.[17] proposed a bio-inspired methodology of biclustering for collaborative filtering that allows overlapping of the biclusters to capture more insightful local structures of the rating matrix. Monika Singh and Monica Mehrotra [59] presented Biclustering Based Collaborative Filtering (BBCF) which used biclustering as a preprocessing step and identified k nearest biclusters for each user to compute interested items for recommendation. Alqadah et. al. [5] proposed a bicluster neighbourhood framework that mapped a user to a bicluster on demand and identified a subset of items from the neighbourhood of the bicluster. Further, the items from the subset were ranked based on the global and bicluster neighbourhood similarity. Kant and Mahara [36] proposed a framework that merged IB and UB techniques in nearest bicluster collaborative filtering method which also allowed overlapping between the biclusters. Xu et. al. [68] solved the problem by using fuzzy c-means clustering technique to identify the correlated subgroup of users based on a subset of items. Later, Bu et. al. [12] extended the framework by analysing the user-item interactions from three different directions. The concept of user-item subgroup was adopted in other variants of recommendation approaches such as location-based [51] as well as for group recommendation [33].

2.2.3 Model-based vs Neighbourhood-based Collaborative Filtering Algorithms

Here we point out some differences between model based and neighbourhood based collaborative filtering algorithms in general.

- Model based algorithms employ various machine learning techniques to build mathematical models by training sample data to make prediction or recommendation whereas neighbourhood based algorithms make predictions or recommendations based on the information of user-based or item-based similarities.
- Model based algorithms require tuning of a large number of parameters based on the problems which makes it a complex method as compared to neighbourhood based algorithms that is easy to understand and implement which can also obtain reasonably good results.
- Model based algorithms are regarded as eager learners as it identifies the hidden associations in the data for training and build a model that doesn't require to go through all the information again in order to make a prediction or recommendation. In contrast, neighbourhood algorithms are lazy learners, means it needs to go through the entire information every time it needs to construct a neighbourhood for a user or item. Thereby, neighbourhood algorithms are slow in prediction computation.

To enhance the performance of traditional collaborative filtering algorithms, we aim to explore powerful search techniques such as metaheuristics in both strategies of collaborative filtering.

2.3 Metaheuristics

Metaheuristic optimization is a stochastic technique that is designed to find a robust, low cost and nearly optimal solution for complex and hyperdimensional problems that cannot be solved with traditional optimization methods. It has been increasingly employed to find quality solutions efficiently for real-world complex problems due to the ability to tackle non-linear and discontinous functions present in such problems. Some researchers have shown substantial advancement in obtaining quality recommendations through the application of metaheuristic techniques in combination with collaborative filtering [6, 19, 22, 47, 66]. These metaheuristic approaches enhance the searching process for finding optimal solutions efficiently. It works on the tradeoffs between local search and randomization. The randomization element in the metaheuristic

technique provides a way to drift the search away from the local best solution so as to avoid from getting trapped into local optima, unlike heuristic approaches. Note that the heuristic techniques do not have the element to get rid of possible trap in local optima. Obtaining an optimal solution in heuristic or metaheuristic framework is not guaranteed all the time. It is more suitable in situations wherein finding optimal solutions is not necessary but finding good solutions are sufficient to solve the problem. Metaheuristic approach is a problem-independent heuristic approach that is applicable even when the problem has partial information. The approach is powerful because of the searching characteristics that focus on exploitation and exploration in the search space. Some of the popular metaheuristic algorithms are genetic algorithm [31], particle swarm optimization [38], ant colony optimization [21], artificial bee colony [37] etc. The following are some of the basic steps adopted to achieve optimal solutions in metaheuristic algorithms:

- Randomized initialization of candidate solutions in hyperdimensional search space.
- Evaluate fitness of each solution in the search space with respect to an objective function derived from problems.
- Navigate the candidate solutions towards global optima in the search space.
- Generate a new population of candidate solutions by performing learning process which is different for every specific metaheuristic algorithms. For example, genetic algorithm learns and upgrade the quality of population through evolution whereas particle swarm optimization does by accelerating the positions of the candidate solutions towards the global solution.
- The algorithms iteratively improve the candidate solutions to generate nearly optimal solutions until stopping criteria is met.

Most of the metaheuristic algorithms have their own structure for guiding the randomly initialized candidate solutions towards the nearly optimal solutions. Note that the fitness function of the metaheuristic algorithms is different for each problem. We employ two popular algorithms, namely, particle swarm optimization and genetic algorithm to address issues in collaborative filtering based recommender systems.

2.3.1 Particle Swarm Optimization

Eberhart and Kennedy [38] introduced *Particle Swarm Optimization (PSO)* in 1995 as a metaheuristic optimization method that simulates the movement of a group of animals. PSO is a metaheiuristic algorithm that imitates a group of flocking birds that has an intelligent communication skill between them. In search of an optimal solution for an optimization problem, PSO adopts the fundamental concept of metaheuristic method mentioned in the previous section. A candidate solution is called as *particle* in PSO. A swarm of particles is randomly initialized in the search space which move around the space in search of an optimal solution. The particles are connected to each other according to a social structure which is referred to as the *neighbourhood*, so that the information collected by each particle is properly communicated among the particles in the neighbourhood. The movement of the particles in the swarm are guided by a *cognitive* component, a *social* component and the *inertia* in the velocity of a particle, which are briefly explained below [7].

- A *cognitive component* of a particle is obtained from the particle's personal best position y. Each particle maintains the best postion visited so far, which is called as personal best position.
- A *social component* of a particle is obtained from the global best particle $\hat{y}$. The postion of the best particle in the swarm that the particle i is aware of is regarded as the global best particle.
- An *inertia component* is obtained from the previous velocity of the particle.

The position of each particle is represented by $particle_i(t)$ where i represents the particle at time step t. The movement of the particle is updated by adding a velocity vector $vel_i(t)$ in the particle's old position wherein the velocity vector drives the particle towards the optimal solution with the influence from social and cognitive components. The position and velocity update rules are given by Eqn (2.14) and Eqn (2.15).

$$particle_i(t+1) = particle_i(t) + vel_i(t+1) \tag{2.14}$$

$$vel_{ij}(t+1) = w.vel_{ij}(t) + c_1 rand_{1,j}(t)(y_{ij}(t) - particle_{ij}(t)) + c_2 rand_{2,j}(t)(\hat{y}_j(t) - particle_{ij}(t)) \tag{2.15}$$

where c_1 and c_2 are acceleration constants that control the rate of participation of social and cognitive components in the velocity update rule. The uniformly distributed random values are represented by the variables $rand_{1,j}(t)$ and $rand_{2,j}(t)$ such that the values range from [0,1], and the inertia component in the velocity update rule is represented by w. The random vectors $rand_1$ and $rand_2$ takes care of the stochastic component in the algorithm. The personal best position y_i of each particle i can be evaluated as in Eqn. (2.16) [7]:

$$y_i(t+1) = \begin{cases} y_i(t), & if \quad f(particle_i(t+1)) \geq f(y_i(t)) \\ particle_i(t+1), if & f(particle_i(t+1)) \leq f(y_i(t)) \end{cases} \tag{2.16}$$

where f represents the objective function of a minimization problem. The global best position, $\hat{y}$, which is the best postion of the particle found so far by the swarm can be evaluated as given in Eqn (2.17):

$$\hat{y}(t) \in y_0(t) \cdots y_{n_s} | f(\hat{y}(t)) = min\{f(y_0(t)) \cdots f(y_{n_s}(t))\} \tag{2.17}$$

Note that the best solution found by the swarm is never lost and is kept track in every iteration. The psuedo code of the PSO framework is presented below:

1. Let D be the number of particles in the swarm.
2. Initialize the position of each particle $particle_i$ to a random vector.
3. If the position of the personal best particle is better than the global best particle ($y_i(t+1) < \hat{y}(t)$), then refresh the swarm's best position.
4. Do until the stopping criteria is met:
   (a) For particles $i = 1, \ldots, D$, do
   (b) Update the position and velocity of the particles by using Eqn (2.14 and 2.15).
   (c) The personal best position of each particle $particle_i$ can be computed by using the Eqn (2.16).
   (d) The global best position can also be computed by using the Eqn (2.17) which selects the best personal best particle.

5. Finally, global best particle is the most optimal solution.

So far we have discussed only about the global best structure of PSO algorithm wherein the whole swarm is one neighbourhood and all the particles are connected to each other [7]. However, in local best structure of PSO the particles are connected and communicated based on the topology adopted. Topology is the social structure of the particles that determines how a particle moves around the search space and which other particles are allowed to be communicated. The topology influences the algorithm in exchanging information of the particles, and hence the rate of convergence depends on it. Some well known topologies are a) Ring Topology b) Star Topology c) Wheel Topology and d) Cluster Topology. The possible termination conditions of PSO algorithm also varies as it can be when 1) the algorithm iterates till a specified maximum number of iteration or 2) when the global best position is not changed for a number of iterations or 3) when a specified elapsed time is reached.

2.3.2 Genetic Algorithm

Genetic Algorithm (GA) [31] was officially introduced by John Holland in 1975 wherein the algorithm simulates biological evolution which is a process of natural selection, to artificial genetic systems to solve real-world complex problems. Genetic algorithm follows the fundamental concept of metaheuristic as mentioned in Section2.3. A candidate solution is termed as *chromosome* in GA. Initially, a population of chromosomes are randomly generated in a search space. In each generation, genetic algorithm selects chromosomes based on a fitness value evaluated and reproduces better solution by performing genetic reproduction operators such as selection, crossover and mutation. Over successive generations, the population converges to global optima. The main operators in the genetic algorithm are listed below:

- Encoding: The chromosome is represented by using an encoding process. The solutions can be encoded in bit string, real numbers, arrays, matrix, trees, list etc, based on the problem.

- Selection: Selection operator chooses a set of parents from the population to perform reproduction operation. Based on Darwin's theory of evolution which is survival of the fittest, best solutions are assumed to survive and generate new

solutions to solve the problem. Some well known selection operators are described below.

- *Roulette Wheel Selection:* It adopts linear search approach in a circular weighted wheel with the aim to find fitter chromosomes for mating. The number of times the wheel is spun depends on the total number of chromosomes in the population. Each chromosome has a slot in the wheel which is selected depending on the wheel's marker in each spin. The selected chromosomes are added to the mating pool for further generation. The variance in the fitness of population controls the rate of evolution.
- *Rank Selection:* Roulette wheel selection will have issues when the fitness value of the best chromosome is too high and occupies almost all face of the wheel. The remaining chromosomes will have less chance to be selected. Therefore, it is avoided by ranking the population based on their fitness.
- *Tournament Selection:* Unlike roulette wheel selection, a number of chromosomes compete with each other to be selected for mating. The competition is based on the fitness value of each chromosome. The winner of the tournament competition is added to the mating pool, and the competition continues until the mating pool is full. The mating pool of tournament selection has larger average fitness population which makes tournament selection more efficient.

- Crossover: Crossover process is a recombination process in the sense that the selected fit candidate solutions are used to generate new solutions.It assumes that recombination of fit solutions has higher chances of producing fitter candidate solutions for further optimization. There are different ways to perform crossover depending on the problem in hand. Some of them are explained below with the assumption that the chromosome is represented as a bit string.
  - *One-Point Crossover:* A point is chosen in the bit string of a parent chromosome and the chromosome is separated into two parts. Two such chromosomes exchange the separated part with each other and then recombines to form two new offspring chromosomes.

  - *Two-Point Crossover:* Two points are chosen in the parent chromosome that separates the chromosome into three parts. After that the contents between the two points in the chromosome is exchanged with another parent chromosome and this leads to the generation of two new offsprings. The advantage of increasing the number of points is to thoroughly search the space though there is a risk of disrupting the good chromosomes in the current population.
  - *Uniform Crossover:* In uniform crossover, a randomly generated binary crossover mask of length equal to the parent chromosome is used to take the decision of recombining two parent chromosomes. Wherever the bit is 1 in the binary mask, the corresponding gene is copied from the first parent chromosome, and when the bit is 0 the corresponding gene is copied from the second parent. The binary mask is newly generated for each pair of parents.
  - *Crossover Probability:* A Chromosome Probability (P_c) is employed to determine how frequently crossover operation will be performed. There is a possibility of no crossover operation and in such cases the parent chromosomes is directly passed as offspring to the next generation for mating. However, it does not imply that the new population is the same, since crossover operation assumes that some good characteristics of old chromosomes passes on to the offspring so that the offspring will be better.

- Mutation: Mutation operator is the one that maintains the genetic diversity in the population by randomly modifying the chromosome. It helps the algorithm to explore the whole search space. It also ensures ergodicity of the search space. If a search space has a possibility of producing new solution from any population state, then it is called ergodic. Some of the mutation operations are described below.

  - *Flipping:* A mutation chromosome is created to determine which bit of the parent chromosome should be flipped to generate offspring. When the bit is 1 in mutation chromosome, the bit in the parent chromosome is flipped to 0 for offspring generation, and when the bit is 0, the bit in parent chromosome is flipped to 1.

- *Interchanging:* Two random points are generated in the parent chromosome and the corresponding bits are interchanged to generate the offspring.
- *Mutation Probability:* A Mutation Probability (P_m) is employed to decide how often the parts in the parent chromosome will be mutated.

The possible stopping conditions of the genetic algorithm are listed below.

- When the GA evolves till the specified maximum number of generations.
- When the specified elapsed time is reached.
- When there is no change in the fitness value.

2.3.3 Combining Metaheuristics with Collaborative Filtering

There have been research that talks about the combination of metaheuristic techniques with collaborative filtering for addressing certain shortcomings of recommender systems. For instance, Shafiq Alam et. al [4, 22] applied Particle Swarm Optimization (PSO) to solve usage-based recommender system. S. Ujjin and P.J. Bentley [64] employed PSO to learn personal opinions of users' and used it for recommending interesting items to the user. Sagarika Bakshi et. al. [6] and Mohammed Wasid et. al. [66] worked on enhancing scalability in collaborative filtering. Diaz-Aviles et.al. [20] used matrix factorisation technique like SVD (singular value decomposition) for factorizing the matrix which unfortunately is not the appropriate technique for factorisation when the matrix has missing values. Assad Abas et. al. [2] presented a thorough survey on how different computational intelligence techniques can tackle various issues related to recommender systems but remained silent on any work incorporating particle swarm optimisation or genetic algorithm with maximum margin matrix factorisation. Two works that are relevant to our idea is the one that incorporates genetic algorithm with MMMF [47] and particle swarm optimisation with MMMF [19]. We found very little research that analyzes the neighbourhood method in capturing the duality between users and items with metaheuristic techniques. As mentioned earlier Pablo A.D. et. al. [17] presented an immune inspired algorithm for biclustering in collaborative filtering. C. Selvi and E. Sivasankar [57] employed a modified cuckoo search technique by combining with fuzzy c-means in recommender system.

2.4 Shortcomings of Collaborative Filtering

Though collaborative filtering approach is more accurate than content based approach for recommender systems [43], the collaborative filtering algorithms suffer from two problems.

- *Scalability:* When we deal with real world data sets for testing the performance or evaluation of a recommender system, the number of users and items present in those datasets in terms of volume is huge. Moreover, only a few users rate the items and that too only a few items are rated. Thus, the benchmark datasets that is used in recommender systems is very sparse and yield poor accuracy. A solution to mitigate this problem would be using techniques such as matrix factorization or use content data in collaborative filtering algorithms.

- *Cold start problem:* The cold-start problem occurs due to the lack of interaction of users/items with the system. For example, a recommender system cannot make any recommendation for a new user in the system as the new user has not made any interactions with any of the items already given. So, collaborative filtering fails to suggest any item that matches with the users' interest. Similarly, an item cannot be recommended accurately to a user since no user has made any interactions to the new item. One of the solutions for the cold-start problem is to exclude those users and items for which there is not even a single rating available.

2.5 Evaluation of Collaborative Filtering Algorithms

In order to evaluate the performance of collaborative filtering algorithms as well as to analyse the performance of collaborative versus non-collaborative filtering algorithms three crucial points need to be examined. They are 1) Datasets 2) Experimental Settings and 3) Evaluation Measures.

2.5.1 Datasets

The collaborative filtering algorithms are evaluated by carrying out experiments on datasets that have preferences of users (usually given as a feedback) over a set of items.

Table 2.1: Statistics related to the datasets being used in this thesis

Dataset	**# users**	**# items**	**# Ratings**
Movielens 100k	945	1682	100,000
Movielens 50k	447	1682	50,000
Movielens 10k	93	1682	10,000
Movielens 1M	6040	3952	1,000,209

The datasets that collaborative filtering algorithms evaluate is usually differentiated on the basis of the feedback as being *implicit* or *explicit*. In the case of explicit feedback, the user consciously discloses his/her preference for an item on a discrete numerical scale. For example, a rating range of $1-5$ indicates that the highest rating 5 for an item is the most liked item and the lowest 1 being the least liked item. The implicit feedback is deduced from users' behaviour such as browsing history, purchase history etc. In our experiments, we use benchmark datasets like Movielens [1] and also extract some smaller datasets from Movielens for further experimental ananlysis. Table (2.1) describes the statistics of datasets based on the number of users and items. The details of the datasets that are being used in our experiments are described below.

- *Movielens 100k:* The Movielens 100k dataset consists of 100,000 number of ratings that ranges from 1 to 5 wherein 1 and 5 indicate the least and most preferred ratings respectively. The missing ratings in the dataset are indicated by 0. The ratings are provided by 943 users on a set of 1682 items. The dataset has some additional movie information such as movie title, release date, IMDB url, genre, video release date and users' demographic information such as age, gender, occupation and zip code. The dataset was debuted in September 1997 from a web-based recommender system. To make the dataset compatible with research experiments, users are selected in such a way that all the users in the dataset have rated atleast 20 items.

- *Movielens 50k and 10k:* The Movielens 50k and 10k datasets are extracted from Movielens 100k dataset and contain 50,000 and 10,000 ratings, thus the datasets have the same rating range. The Movielens 50k dataset has 447 users rated on 1682 items while the 10k dataset has 93 users rated on 1682 items.

- *Movielens 1M:* The Movielens 1M dataset has 1,000,209 number of ratings provided by 3952 users on a set of 6040 items. The ratings range from $1 - 5$. The Movielens 1M dataset also has same additional information as in Movielens 100k dataset.

2.5.2 Experimental Settings

In this section, we explain some of the intricacies involved in evaluating the performance of recommender system algorithms and outline certain steps that needs to be taken for a fair comparison. In order to compare the performance of two different recommender system algorithms, it is necessary to fix controlling variables of the algorithms. For example, if we want to compare algorithm P and algorithm Q with respect to prediction accuracy of missing ratings, and given that the algorithms P and Q are trained on different datasets, then it is difficult to conclude as to which algorithm performs better. A solution to this problem is to train both the algorithms with the same dataset or train the same algorithms on different datasets. The benefits and drawbacks of popular evaluation approaches are explained below [58].

1. Offline Experiments:

An offline experiment is executed on pre-collected dataset that has users' opinions over several items. The researchers use offline experiments extensively in recommender system to evaluate performance of the algorithms and compare among them since the offline experiments are simple and low cost. In offline experiments, it is assumed that users' future actions can be interpreted from past users' actions through simulation to make a reliable suggestion. Moreover, offline experiments are more suited for evaluating prediction accuracy of algorithms as it analyses quantitative aspects rather than subjective aspects as in the case of online experiments. To perform offline evaluation, data is partitioned into two sections, namely *training data* and *test data*, wherein the partitions are performed in following ways:

- *Holdout:* Test set is constructed by randomly selecting a group of existing ratings from the entire dataset, and the rest of the ratings make the training set. For example, test set can be constructed by randomly selecting 20% of the existing ratings in the dataset and the training set is formed by the rest 80% ratings of the dataset.

- *User-based Holdout:* In this case, a fixed percentage of existing ratings from a user's past activity is randomly selected to construct the test set and the remaining ratings form the training set. For example, 20% of the ratings provided by a user is randomly selected as the test set and the remaining 80% ratings as the training set.

Once the dataset is partitioned into test set and training set, the test set data is used to cross check the prediction accuracy or relevance of the recommendations made by the algorithms whereas the training set is used to make the algorithm learn to make accurate predictions or recommendations. There are a few benefits and drawbacks of employing offline experiments in evaluating recommender system algorithms.

Benefits:

- Offline experiments need not have any association with the real users and therefore allows it to make an affordable comparisons among several recommender system algorithms as it is cost-effective.
- Offline experiments are suitable for evaluating quantitative aspects of the algorithms such as prediction or recommendation accuracy.

Drawbacks:

- Offline experiments are restricted to quantitative analysis of recommender systems in terms of prediction or recommendation accuracy as real user associations are not available. It cannot evaluate the algorithms in terms of subjective aspects such as measuring how much the recommended list of items influence the users or how much the recommended list is satisfactory etc.
- Offline experiments cannot expand possible changes in the preferences of real users such as change in the users' preferences over time since it assumes that the past behaviour moulds the future behaviour. Therefore the recommended list of items through offline experiments may not satisfy the user.

Thus, offline experiments are usually employed to select a small set of appropriate algorithms mostly related to test costly user-studies. They are also used as part of online experiments for tuning the parameter values and after fixing the parameter values in

an offline mode, online experiment are carried out on the best tuned parameter of the algorithm to make recommendations to real users.

2. User Studies:

Several recommendation algorithms rely on users' association with the system and therefore predicting a reliable association of users with the system is difficult. Some researchers [40, 50] conduct user studies by hiring a set of test subjects and provide them a recommended list of items to analyze their interactions with the recommended list. While the subjects under test are getting associated with the system, their behaviour is observed and analyzed to interpret subjective aspects such as whether the recommended list was accurate with respect to the user or how much time did a particular user spend on the recommended items or whether the test subjects were surprised to see the items in the list etc. The test subjects are also provided questionnaires or are surveyed to collect their opinions on the recommendations to get qualitative information.

Benefits:

- It is easy to observe influence of recommendations on users in user studies.
- Obtaining qualitative information is easy in user studies

Drawbacks:

- Though analyzing the users' behaviour directly is easy to get qualitative information, it is very expensive to execute the approach in terms of time or cost if the test subjects are not volunteers. The approach has to be repeated as well to make a reliable conclusion.
- If the test subjects do not represent the major pool of users in the real system, there is a possibility that a kind of bias occurs in favour of a particular set of users. When the test subjects are aware that they are involved in an experiment especially in the case of paid subjects, it is most common that they are biased in favour of the company conducting the experiment.

3. Online Experiments:

As mentioned earlier, offline experiments cannot determine correct influence of the recommended list of items to a user unless the items are already consumed by the

user. Note that a recommended item is determined irrelevant to the user in offline experiments if the item is not included in the test set. However, the recommended item may be relevant to the user but the offline experiment declares as irrelevant since the item is not consumed by the user yet. It may be due to a scenario that the user is not aware of the item. Also, offline experiment cannot achieve a true feedback from user if the recommended item is unknown to the user. The possible changes in users' preferences over time also cannot be obtained in offline experiments. To tackle the problems mentioned, online experiments are executed wherein the users interacts with the system in real time.

With online experiments, the subjective information of a user's behaviour on a recommended list of items is easily achieved. The subjective information could be the amount of time a user spends on an item that is there in the recommended list or it could be related to the newness/diverseness of the items in the recommended list. Online experiments can also help in ranking the algorithms in the sense that a few candidate algorithms can be compared to find which one is superior than the others. It is a common observation in the recommender system community that the algorithm that the users follow more frequently are determined to be superior than that of others. Thus, several real world recommender systems perform online experiments to compare multiple algorithms [41] to evaluate the performance by using user-centric metrics such as number of views or number of clicks on a page etc. [24]. The random user selection is the key to perform a fair comparison in such experiments.

Benefits:

- Online experiments capture the change in user's preferences over time.
- Understanding the reason behind the users' preference of an algorithm over others leads to improving the quality of the recommender system. In other words, user satisfaction in the recommended list of items determine the quality of recommender system algorithms. Online experiments gives the access to evaluate the user satisfaction of a recommender system.
- It is the only experiment that performs on direct interaction of real users with the system. It can also evaluate overall system objective such as long term profit or users' continued interest in the recommended items and how these system objectives depend on accuracy and diversity of the recommendations.

Drawbacks:

- Online experiments are expensive when multiple algorithms are there to compare.
- Online experiments are risky if the recommendations made in the past are not accurate to the users' interest. It may even have a negative impact which is undesirable in e-commerce platform.

Therefore, it is suggested to perform online experiments with a few candidate algorithms after an extensive offline experimentation to conclude the quality of a candidate recommendation algorithm.

2.5.3 Evaluation Measures Discussion

The recommender system algorithms have several aspects to be considered while evaluating the quality of the algorithm such as the task for which the algorithm is designed, the type of data the algorithm is designed to work with, etc. In the literature, several researchers proposed multiple evaluation measures to judge the performance of recommendation algorithms according to different aspects. In recommender systems, the evaluation measures are employed to analyze the performance of the candidate algorithms and compare with the performance of existing algorithms. Based on the different perspectives of the recommender systems, evaluation measures are categorized into two, namely *Accuracy measure* and *Beyond accuracy measure*.

1. Accuracy Measure:

The accuracy measure is the most popular measure used in recommender systems as can be seen from the literature. Most of the researchers aim at evaluating the accuracy of the recommendation algorithms because the prime objective of any recommender systems is to either carry out prediction or recommendation. Thus, the task of the recommender systems is categorized into two sections, namely *Prediction Task* and *Recommendation Task*, for which different accuracy metrics are proposed [29, 58].

- *Prediction Task:* A recommendation algorithm that generates accurate prediction of missing ratings is considered as an ideal algorithm in recommender systems technology since the fundamental assumption is that the users' prefer accurate

predictions. Therefore, most of the recommendation algorithms adopt this assumption to predict ratings to a value that is close to the true ratings. Once the rating prediction is completed, the algorithm can be evaluated with respect to the predictive ability by using predictive accuracy metrics.

Most of the ratings in recommender systems are in numerical range. The predictive accuracy metrics for numerical range evaluates how far the predicted ratings is from the true rating by evaluating the difference between the two ratings. There are two well known predictive accuracy metrics, namely *Mean Absolute Error (MAE)* and *Root Mean Squared Error (RMSE)*, to evaluate the performance of a recommender system.

- *Mean Absolute Error (MAE):* MAE evaluates the average deviation of predicted rating from the corresponding true rating. It is described by Eqn. (2.18) where t corresponds to test ratings, Y_i corresponds to true ratings and X_i denotes the corresponding predicted rating:

$$MAE = \frac{\sum_{i=1}^{|t|} |Y_i - X_i|}{|t|} \tag{2.18}$$

- *Root Mean Squared Error (RMSE):* RMSE squares the deviation between the predicted rating and the corresponding true rating and takes the squared root of the average value. It is a metric that penalizes large errors. It is described by Eqn. (2.19)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{|t|} (Y_i - X_i)^2}{|t|}} \tag{2.19}$$

Both the RMSE and MAE metrics are not suitable for top-N recommendation task since these metrics compute errors for all the test ratings including the items that are not included in top-N list. The users are not concerned about the errors of other items which are not in top-N list. The difference between the two metrics is that the RMSE would determine an algorithm A better when comparing with an algorithm B if the algorithm A has smaller error over several test items whereas MAE would determine that algorithm B is better since algorithm B has larger error on the test items.

- *Recommendation Task:* Recommendation task suggests a list of possibly interested items to a user which is the final goal of any recommender systems. Thus, recommendation task requires to identify correct relevant items which can be achieved by following two popular measures, such as *Precision* and *Recall*.

  - *Precision:* Precision measures how many of the top N recommended list of items are relevant to the user. It is computed as a fraction of the relevant items among all the retrieved items over the number of retrieved items.

$$Precision = \frac{relevant \cap retrieved}{retrieved} \tag{2.20}$$

  - *Recall:* Recall measures completeness of the recommendation. It is computed as the fraction of relevant items among all the retrieved items over the number of relevant items.

$$Recall = \frac{relevant \cap retrieved}{relevant} \tag{2.21}$$

  - *F1-score:* F1-score computes the harmonic average of the precision and recall.

$$F1score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.22}$$

  - *Mean Average Precision (MAP):* Mean Average Precision measures the quality of recommendation across the levels of recall. When each item in the top N list is correctly recommended, average precision is computed at each correctly recommended item for each user j. The average precision calculated at each correctly recommended item is again averaged over the set of users U where R_{jk} is the top N results and m_j is the number of correctly recommended items [68].

$$MAP(U) = \frac{1}{U} \sum_{j=1}^{|U|} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk}) \tag{2.23}$$

  The precision and recall are biased on the number of top N recommendations. For example, if the number of retrieved items is much larger than the number

of relevant items, then the precision value becomes very small. Whereas if the number of relevant items is larger than the number of retrieved items, then we will get low recall value. The F1-score value is more reasonable when some users have more ratings in the test set than others.

2. Beyond Accuracy

- *Coverage [25, 35, 58] :* Coverage of a recommendation algorithm reflects the wider range of items recommended to the users. A high coverage of a recommendation algorithm may make the user trust the system. It measures the degree to which top N recommended list covers the total number of items.

$$Coverage = \frac{|I_{pr}|}{|I|} \tag{2.24}$$

  where I_{pr} represents the group of predicted items and I represents the total number of items

- *Novelty [14, 35]:* Novelty of a recommended item in RS corresponds to how different the item is and need not necessarily be compared only with the set of recommended items. The items with less number of ratings are likely to be unknown to a user and recommendation of such items are considered as novel.

- *Unexpectedness [3]:* The concept behind unexpectedness is to recommend some items that the user is unaware of. Though the definition of unexpectedness and novelty is overlapping, they are distinguished in the sense that the recommended items should be relevant but different from the user's expectations whereas the novelty of a recommended item is not defined with respect to users but with the rest of the items.

- *Serendipity [35, 44] :* Serendipity is defined by the degree of interestingness and unexpectedness of a recommended item.

- *Diversity [14, 35, 58]:* Diversity refers to a set of recommended items wherein the items are different from each other. Unlike the measures mentioned above which compute the metrics on a single item included in top N recommendation, diversity is computed based on a set of recommended items.

- *Confidence [58]:* Confidence measures the degree of uncertainty in the prediction/recommendation. A user may utilize the service of the recommender system only based on the level of confidence attached to the items.

2.6 Summary

In this chapter, we have presented a brief introduction to the concept of recommender systems and explained the strategies required to solve the issues that plague such systems. Among the several strategies that have been put forward for building a recommender system like content based algorithms, collaborative filtering algorithms, hybrid algorithms, our aim is to explore the collaborative filtering strategy as it needs only rating data and do not need any additional information about users or items. We have briefly discussed the approaches of collaborative filtering, such as model-based and neighbourhood based methods. There are several similarity measures to find neighbours of an active user in the neighbourhood method. We have described three such methods and then discussed the importance of partial matching of preferences among the users and items in neighbourhood method. Discovering such partial matching of preferences among users and items by using different techniques is one of the contributions of our research.

The role of metaheuristic optimization techniques is discussed and two population based search algorithms are explained which are successful in solving real-world complex problems. Our objective is to explore the powerful metaheuristic optimization algorithms by combining with collaborative filtering since the metaheuristic algorithms are good at finding optimal solutions even when only partial information about the problem is available. In collaborative filtering, only partial information about the user/item is available as the rating data given in the form of data sets is really sparse and no other information about the problem is available. To achieve this end we focussed on two metaheuristic algorithms such as PSO and GA which is to be used in our research. Finally, we have outlined the various evaluation strategies needed for recommender systems and also described the datasets used in our experiments. Several of the evaluation approaches as mentioned in the literature is discussed including offline experiments, online experiments, user studies, and we also talked about their benefits and drawbacks. In order to evaluate recommender system algorithms we also

need specific metrics like precision and recall and we also discussed about various such metrics available in the literature.

Chapter 3

Metaheuristics for Matrix Factorization

In the previous chapter we explained in depth about the various techniques being used in building recommender systems and gave pointers on how a combined approach of collaborative filtering and metaheuristic approached would fetch better results in terms of recommendation accuracy. In this chapter, we propose a combined framework for recommender systems based on collaborative filtering strategy by initially combing it with a metaheuristic approach like Particle Swarm Optimisation(PSO). We make use of the maximum margin matrix factorisation (MMMF) technique for collaborative filtering and combine it withe three variants of particle swarm optimisation, namely, gradient based particle swarm optimisation (GDPSO-MMMF), mutation based particle swarm optimisation(PSOm-MMMF) and hierarchical particle swarm optimisation (HPSO-MMMF). Thereafter we combine MMMF with genetic algorithm (GA) and outline the experimental results related to each one of the proposed method.

As mentioned in the previous chapter, maximum margin matrix factorisation is a very popular technique for collaborative filtering wherein the rating matrix is discrete valued and consists of only a small portion of the known ratings. The main idea is to factorise the matrix into two latent factors (user latent factor/item latent factor) so that the unknown ratings can be estimated form the product of the factors. The user and item latent feature vectors consists of the degree of interest the user or item has for the hidden latent features. For example, the user latent feature vector corresponds to the preferences that the user has for the features of movies like comedy, history, musical,

war etc. Similarly the item feature vector corresponds to the weightage of the features that the movie has. In order to achieve the factorisation a loss function is optimised wherein *gradient descent* or its variants are made use so that a near optimal solution is obtained irrespective of the starting point. The metaheuristic algorithms such as PSO and GA are *exploratory* in nature which enhance the traditional model-based collaborative filtering techniques like MMMF with *exploitatory* nature of gradient descent. The gradient descent approach may get trapped in local optima which is why we plan to employ metaheuristic techniques. Our algorithm starts from multiple initial points and uses gradient information and swarm-search as the search progresses. We show that by this process we get an efficient search scheme to get near optimal point for maximum margin matrix factorization.

3.1 Proposed Algorithms

A mentioned earlier, metaheuristic approaches such as Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) are popular population based search optimization algorithms. In such algorithms, multiple possible solutions are initially randomly generated and scattered in a large search space. PSO algorithm has the potential to find the nearest optimal solution by exploring different regions of the search space and gather information from each particle (solution). The particles in a swarm act in a coordinating manner and make the algorithm succeed in finding a nearest optimal solution over subsequent iterations. Genetic algorithm on the other hand finds the nearest optimal solution through evolution of a set of fittest chromosome by executing the genetic operations.

Here, we combine the population based techniques, such as PSO and GA with MMMF and propose four algorithms. Each proposed algorithm has a benefit over the other in terms of finding the nearest optimal solution. In the first proposed algorithm, we initiate our work with a typical PSO and integrate it with MMMF to find the optimal factored matrices U and V. In the second proposed algorithm, we optimize the previous framework by adding an operation called *mutation* which improves the swarm diversity and this results in a better framework for finding the most optimal solution, i.e. the low rank factored matrices U and V. In the third algorithm, we propose

a heirarchical PSO based MMMF in which a social structure of the particles is constructed based on a hierarchical tree structure wherein a neighbourhood is formed by a parent with immediate children such that the parent becomes the local best particle in the neighbourhood and the root of the tree represents the global best particle. Finally, we explore GA with MMMF that differentiates the search of optimal solution from the previous techniques like PSO. In MMMF, the objective is to find two factored matrices, U and V such that UV^T is as close to the true rating matrix, Y. To find the most approximate factored matrices from a random set of matrices is a complicated problem as the search space is huge. The idea here is to use meteaheuristic approaches in an intelligent manner so as to reduce the search space. The benefit of using metaheuristics approach in this problem domain is based on the possibility of starting the search from multiple possible points in the search space. In this way, the search for the optimal factored matrices is optimized.

A candidate solution of the swarm or genetic population is termed as a particle or chromosome respectively. The candidate solutions of the swarm/population is a user-item rating matrix $X = UV^T$ wherein the best candidate solution would be the optimal solution with the least fitness value. The fitness function of the proposed algorithms is the objective function of MMMF given by Eqn (2.5). The optimal solution is the low ranked factored matrices UV^T that possesses the least loss value since the fitness function is a minimization problem. We represent the candidate solution (particle/chromosome) as a single matrix P that is composed of the factored matrices U and V. The columns of the particle/chromosome P are the user latent vectors $[U_{1,1}, \cdots, U_{f,1}]^\top$ to $[U_{1,n}, \cdots, U_{f,n}]^\top$ concatenated with columns of item latent vectors $[V_{1,1}, \cdots, V_{f,1}]^\top$ to $[V_{1,m}, \cdots, V_{f,m}]^\top$ as shown in (3.1), where the user vector $[U_{1,1}, \cdots, U_{f,1}]^\top$ denotes the preferences of the user u_1 for the latent hidden features of size from $1, 2, 3, \cdots, f$. Similarly the item vector $[V_{1,1}, \cdots, V_{f,1}]^\top$ denotes the degree that the item i_1 has for the hidden latent features of size from $1, 2, 3, \cdots, f$. Note that there are f hidden latent features. The potential solution is given by the Eqn (3.1).

$$\text{Potential Solution, } P = \begin{bmatrix} U_{11} & \cdots & U_{1n} & V_{11} & \cdots & V_{1m} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ U_{f1} & \cdots & U_{fn} & V_{f1} & \cdots & V_{fm} \end{bmatrix} \tag{3.1}$$

The potential solution P equates an approximated user and item factored matrices U, V

such that the dot product of U, V should be close to the true rating Y, given by Eqn(2.1) i.e. $Y \approx UV^T$. The quality of the potential solution P is evaluated by using the optimization function of MMMF given by Eqn (2.5) that uses hinge loss function with a regularization term. The potential solution P is learned by using the gradient descent defined by MMMF in the Eqn (3.2). The searching operation of the swarm/population continues until the optimal potential solution P is found. The optimal solution is the low rank factored user and item matrices that best approximates the true rating Y.

We proposed four matrix factorization based algorithms that are integrated with different search techniques of the metaheuristics approach. The difference among all the proposed algorithms is the searching behaviour of the optimal solution by different heuristic approaches. The common components among the proposed algorithms are listed below:

- The potential solution of *MMMF* is represented as a concatenated matrix of user and item latent vectors represented by $[U_{1,1}, \cdots, U_{f,1}]^\top$ to $[U_{1,n}, \cdots, U_{f,n}]^\top$ and $[V_{1,1}, \cdots, V_{f,1}]^\top$ to $[V_{1,m}, \cdots, V_{f,m}]^\top$ respectively, as shown in Eqn (3.1).
- The fitness of the potential solution is evaluated by using the optimization function of MMMF.
- The potential solutions are learned by using the gradient descent component of the MMMF.
- The optimal solution obtained from the metaheuristics technique is the user-item factored matrices that best approximates the true rating.

3.1.1 Gradient Descent Particle Swarm Optimization based MMMF (GDPSO-MMMF)

In this proposed method, a basic local best structure of PSO is adopted and combined with MMMF to enhance the search of the optimal low rank factored matrices U and V. The initial neighbourhood of PSO component is created randomly and the fitness of each particle is evaluated and circulated. The particle with the best fitness in a neighbourhood is selected as a local best particle that guides the remaining particles in the neighbourhood. The local best particles of all the neighbourhood share their fitness

value and the one with the best fitness value is selected as the global best particle of the swarm. Eventually all the particles in a swarm are gradually directed towards the globally fit particles which are defined by the global best particle. In the proposed algorithm, the velocity factor in the position update rule of PSO is replaced by the gradient direction. The new position update rule of U type of particle is given by Eqn (3.2).

$$U_i^{t+1} = U_i^t - c(\frac{\delta J}{\delta U} - (1-\delta)(U_{best}^t - U_i^t)) \tag{3.2}$$

The updation rule for V, Θ are similar to U. The range of δ is [0,1]. It can be seen that if $\delta = 1$, then the method is the same as applying only gradient descent. On the other hand when $\delta = 0$, the above equations are not equivalent to PSO search. The flowchart of the proposed algorithm is presented in figure (3.1) while the algorithm is stated in the Algorithm (1).

Algorithm 1: GDPSO-MMMF

Input: Matrix Y, partially observed known rating matrix.
Output: Matrix X, fully observed predicted rating matrix.
Initialization of Swarm (Initializing U, V and Θ matrices);
while *(stopping criteria is not met)* **do**
    Evaluate fitness of the particles in swarm using $\mathcal{J}$ function given in equation (2.5);
    for *each neighborhood* **do**
        Find `local best`;
        Update several particles with some probability, ρ, using equation(3.2);
    end
    Find `global best` (gb) particle of the swarm;
end

3.1.2 Particle Swarm Optimization with Mutation based MMMF (PSOm-MMMF)

The previously proposed algorithm GDPSO-MMMF keeps track of the historical optimal position of the swarm. If the optimal position of the swarm does not change for a number of iterations, then there is a possibility of having new optimal solution outside the radius of the swarm and it may not be possible to find the new optimal solution due

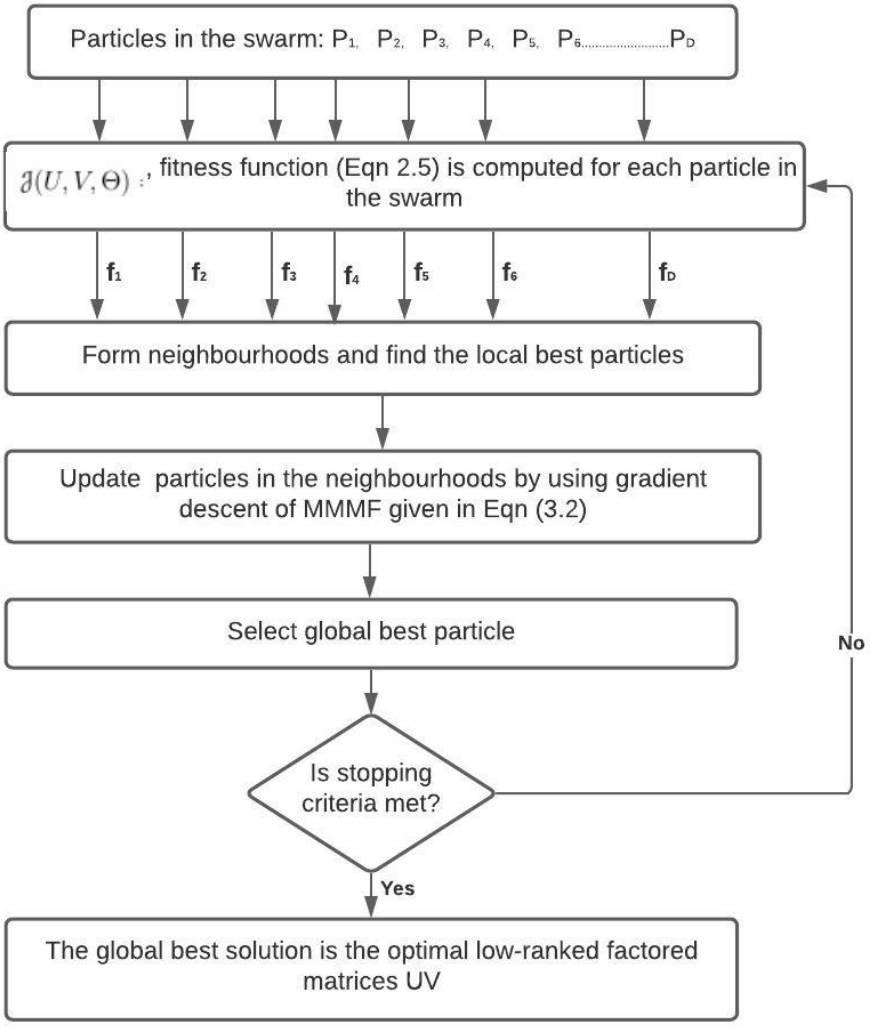


Figure 3.1: GDPSO-MMMF framework

to the loss of diversity. Thus, we propose a modified algorithm that considers the odds by including a component called *mutation* which increases the swarm diversity by re-randomizing some particles of the swarm. The re-randomizing component increases the diversity of the swarm and also maintains the memory of the swarm by preserving some good particles near the changed optimum [7]. Thus, the proposed algorithm PSOm-MMMF extends the previous algorithm GDPSO-MMMF by having a mutation component which enhances the exploratory nature of the algorithm and blocks from possible premature convergence. If the optimal solution of the swarm does not change for a significant number of consecutive times, *MaxSame*, and the radius of the swarm is below a specified threshold, *ThresRadius*, then the position and velocity of the several particles are re-randomised according to some mutation probability ρ, [7]. The radius of the swarm, *SwarmRadius*, is defined by Eqn (3.3).

$$SwarmRadius = max_{j=1...k} \left(\sqrt{\sum_{d=1}^{D} (P_d - lbest_j)^2} \right) \tag{3.3}$$

where D represents the number of particles in the swarm, k represents the number of neighbourhoods, and $SwarmRadius$ corresponds to the maximum Euclidean distance between all the particles, P, and the historic optimal position of the neighbourhood, $lbest_j$.

The flowchart of the proposed algorithm is presented in figure (3.2) for a better understanding of the algorithm and to explain the differences with the previous algorithm *GDPSO-MMMF*. To make sure that the good particles don't leave the interesting regions of the swarm, a threshold, $ThresRadius$, is employed in the proposed algorithm. The proposed algorithm (2) terminates when the mutation occurs for a $MaxSame$ number of times that is set to 5 with no improvement further. The mutation is performed in a probabilistic manner.

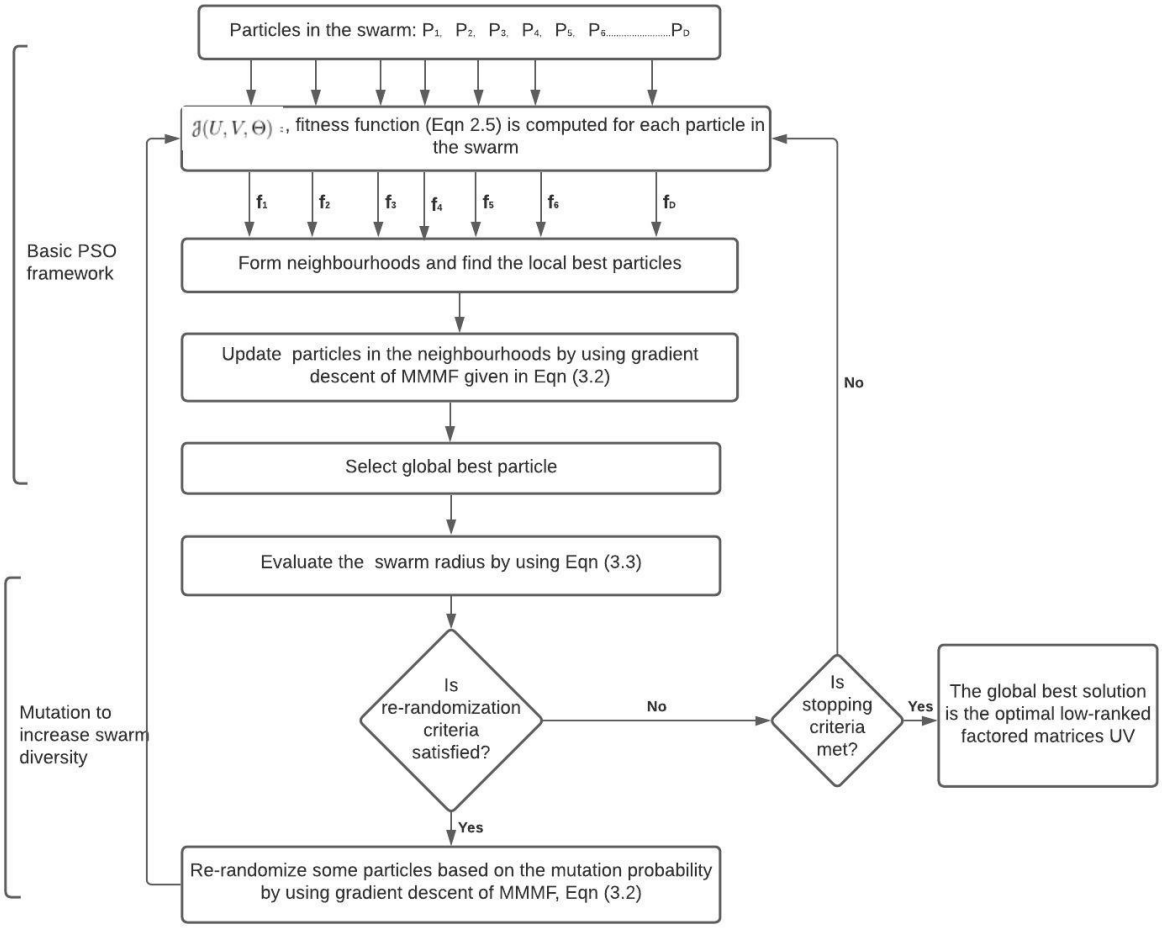


Figure 3.2: PSOm-MMMF framework

Algorithm 2: PSOm-MMMF

Input: Matrix Y, partially observed known rating matrix.
Output: Matrix X, complete predicted rating matrix.
Initialization of Swarm (Initializing U, V and Θ matrices);
while *(True)* **do**
  Evaluate particles in swarm using $\mathcal{J}$ function given in equation(2.5);
  for *each neighborhood* **do**
    Find `local best`;
    Update several particles with some probability, ρ, using equation(3.2);
  end
  Find `global best` (gb) particle of the swarm;
  Evaluate *SwarmRadius*;
  if *gb repeats for MaxSame number of consecutive times and SwarmRadius is less than a threshold,* ThresRadius **then**
    Re-randomize the position and velocity of some particles with the mutation probability ρ ;
    Update the new particles using equation(3.2);
  end
end

3.1.3 Heirarchical Particle Swarm Optimization based MMMF (HPSO-MMMF)

In HPSO-MMMF, we integrate the neighbourhood topology in the search space by ordering the particles structurally rather than assigning particles randomly as in the previously proposed algorithms GDPSO-MMMF and PSOm-MMMF. The particles are ordered in a hierarchical tree structure where a parent with their immediate children forms a neighborhood, with the parent being the best particle in each neighborhood [23]. Each local best particle of a neighborhood also belongs to an immediate upper level neighborhood as a child. The tree structure is shown in Fig (3.4). Rearrangement of the particles in the tree is performed in each iteration such that it guarantees to maintain the global best particle at the root while still maintaining the constraint that the local best particle be the parent in each neighborhood. In this framework, this model ensures that the best particle of a neighborhood influences the other neighborhoods with which it is associated and the best particle of the swarm globally influences

the rest of the particles in the swarm. Thus, it is possible to obtain the optimal solution faster. The proposed algorithm is given in Algorithm (3) and the flowchart in Fig.(3.3). The algorithm terminates when either maximum iterations surpass a user defined threshold or the global best does not improve for a certain number of iterations.

Algorithm 3: HPSO-MMMF

Input: Matrix Y,partially observed known rating matrix.
Output: Matrix X, complete predicted rating matrix.
Initialize swarm (U, V and θ matrices);
Create tree with the particles;
while *(True)* **do**
    Evaluate fitness value for each particle using equation (2.5);
    Rearrange the tree with parents having better position than their children in a neighborhood;
    Update U^t_{best}, V^t_{best}, θ^t_{best};
    Update U^{t+1}, V^{t+1}, θ^{t+1} using (3.2);
end

3.1.4 Genetic Algorithm based MMMF

In GA-MMMF, we incorporate the genetic algorithm with MMMF with an aim to exploit the survival of the fittest approach of the genetic algorithm. The possible solutions, namely chromosomes, are encoded in value encoding. The representation of a chromosome is shown by Eqn. (3.1) such that the values in U part corresponds to the degree of preference the user has with respect to the corresponding features whereas the values in V part shows the degree of a feature the movie has. In Crossover genetic operator, two points are randomly generated in the chromosome where one point lies in the U component and the other one in the V component. The two points are selected in the above manner to make sure that the factor matrices U and V are fully engaged in the genetic operations. In Mutation operator, some randomly selected chromosomes based on mutation probability of the population are updated by using gradient descent as in Eqn (2.7). For simplicity, we have presented the flowchart of the proposed algorithm in Figure (3.5).

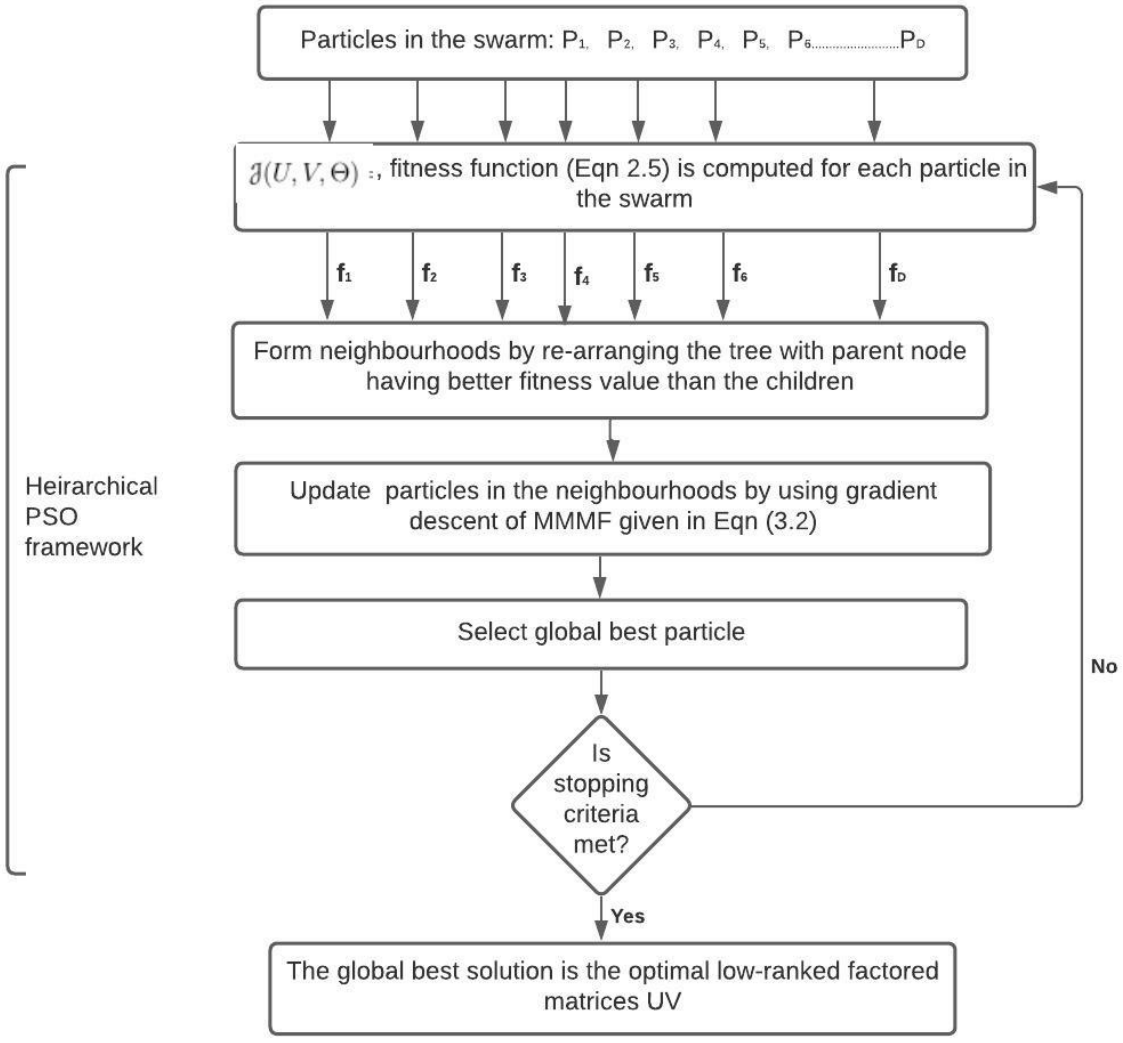


Figure 3.3: HPSO-MMMF framework

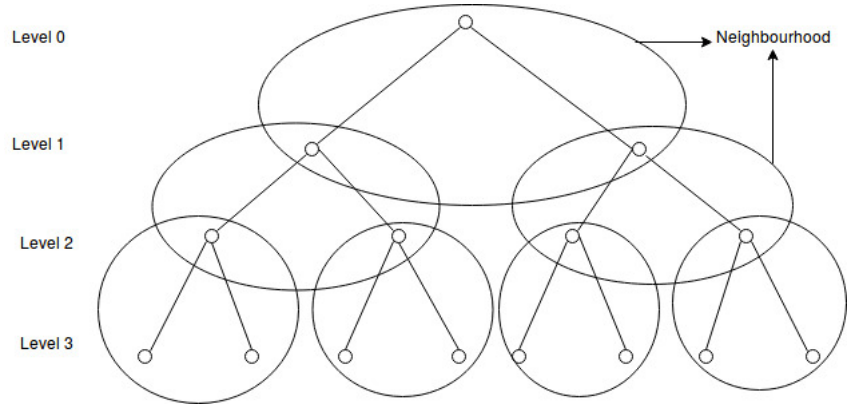


Figure 3.4: Heirarchical Tree Structure of neighbourhood topology

Algorithm 4: GA-MMMF

Input: Matrix Y,partially observed known rating matrix.

Output: Matrix X, complete predicted rating matrix.

Initialize population (U, V and θ matrices) ;

Evaluate the fitness values of the chromosomes using equation (2.5);

while *(True)* **do**

    Select parents using tournament selection operator;

    Reproduce offsprings using crossover and mutation operator. Apply gradient descend given in equation(3.2) in mutation;

    Selection for the survival of the fittest;

end

3.2 Experimental Settings and Results

Experimental analysis of our proposed algorithms has been carried out on a benchmark dataset called Movielens. Movielens consists of 100,000 ratings in the range of 1 to 5 such that the least number, 1, shows the least preference while the largest number, 5, shows the highest preference. The missing ratings are represented with a 0 entry. We also conducted experiments on smaller datasets such as 10k and 50k ratings, extracted from the 100k ratings Movielens dataset. The movielens dataset is constructed in such a way that each user has rated at-least 20 movies. The dataset was collected by

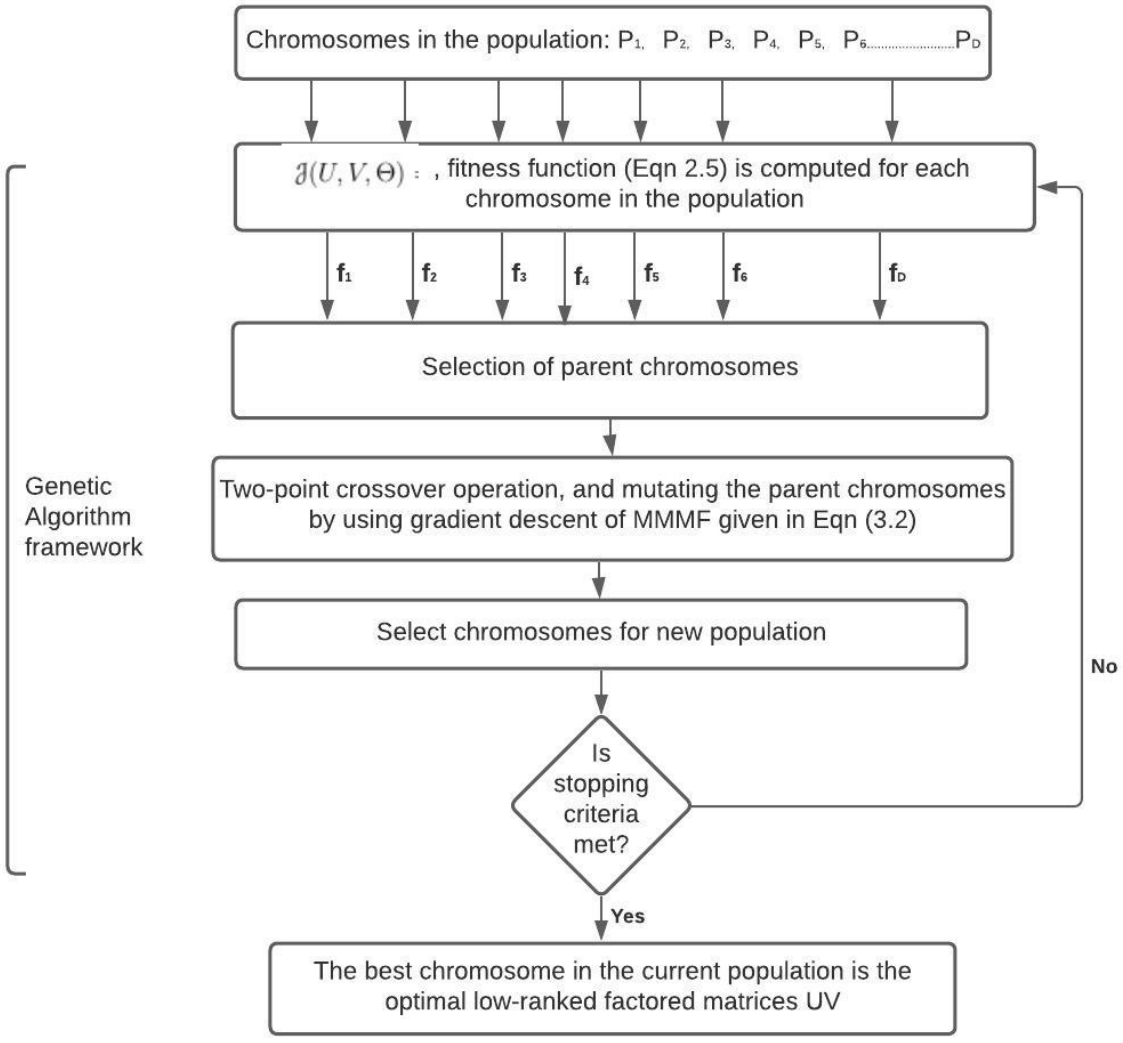


Figure 3.5: GA-MMMF framework

University of Minnesota under the project name Grouplens [1]. The experiments were performed on Microsoft Windows 8, 64 bit Intel Core i3 and Matlab tool was used. In our experiments, we split the data randomly into two partitions, such that 80% forms the training set and 20% the test set. Since we are concerned with improving the accuracy in the rating predictions, we used popular metrics such as *Root Mean Squared Error (RMSE)* and *Mean Absolute Error (MAE)*. As is well known, smaller values of the accuracy metrics represent better performance of the algorithm.

3.2.1 Parameters Settings

In the proposed algorithms, the population size, D, is set to 50. The genetic probability is set to 0.7 so that when the genetic probability is below 0.7 crossover operation is executed and when it is above 0.7 mutation operation is executed. Experimental ananlysis of the proposed algorithms were carried out with different number of latent features f such as 10, 50 and 100, to fix it at a value that gives the most accurate result. The value of f was set to 100 and further experiments were conducted with this fixed value. The size of the particle/chromosome, P, is the total number of users and items, and f, is the number of columns and rows respectively. For example, the size of particle/chromosome, P, for 100k dataset is $943 + 1682 = 2625$ and 100 is the number of columns and rows. In the proposed algorithms (1 and 2), the number of particles to be updated are defined by multiplying the probability ρ with the population size wherein the parameter ρ is set to 0.3. The parameter $ThresRadius$ which is used to indicate acceptable radius of the swarm is set to 100.

3.2.2 Results and Analysis

The results of the proposed algorithms and comparison of the proposed algorithms with the base algorithm (MMMF) are presented in figures (3.8,3.7,3.6) and tables (3.1,3.2). The figure 3.8 presents the graphs comparing the results of all the proposed algorithms, in terms of RMSE and MAE, with respect to the base algorithm across a number of generations. We can see from Fig. (3.8) that the results obtained by the heuristic search based MMMF outperform the traditional MMMF. Among the proposed models, the results obtained by GDPSO-MMMF, PSOm-MMMF and HPSO-MMMF are better than

[1] http://grouplens.org/datasets/movielens

GA-MMMF as the GA based algorithm took longer time to converge to a nearly optimal solution as compared to the PSO based proposed models. We observe from Fig. 3.8 that the proposed model, PSOm-MMMF, starts achieving a lower error rate from early generations as the re-randomizing component of the model is able to capture the nearly optimal solution from early generations. All the three variants of PSO based proposed algorithm ultimately converge to a similar optimal solution while the difference being that which model converges faster. Similar behaviour is observed on the 10k (Fig. 3.6) and 50k (Fig. 3.7) MovieLens dataset with all the proposed algorithms with some exception of premature convergence.

Tables 3.1 and 3.2 present results related to RMSE and MAE values of our proposed algorithms as compared against the base algorithm(MMMF) and another closely related work[19] on benchmark Movielens datasets of 10k, 50k and 100k. The results depicted in the tables show that the proposed combination of population based search and matrix factorisation method performs better than the base MMMF model. We sometimes observe similar performance from the proposed algorithms PSOm-MMMF and GDPSO-MMMF as can be seen from the experimental results. The reason of obtaining similar performances of the two proposed models is that even though we randomly re-initialized several particles based on the satisfactory criteria of mutation operation, the fitness of the best solution of the previous iteration is better than newly initialized solution of the new search space. In Tables 3.1 and 3.2, we also compared our proposed algorithms with an earlier work [19]. We observe that the performance of our proposed algorithms are better as compared to that of [19] as our proposed algorithms have more advanced framework such as *neighbourhood topology*, *PSO based mutation operation* and *Genetic Algorithm* as compared to the simple global best framework of PSO being combined with MMMF in [19].

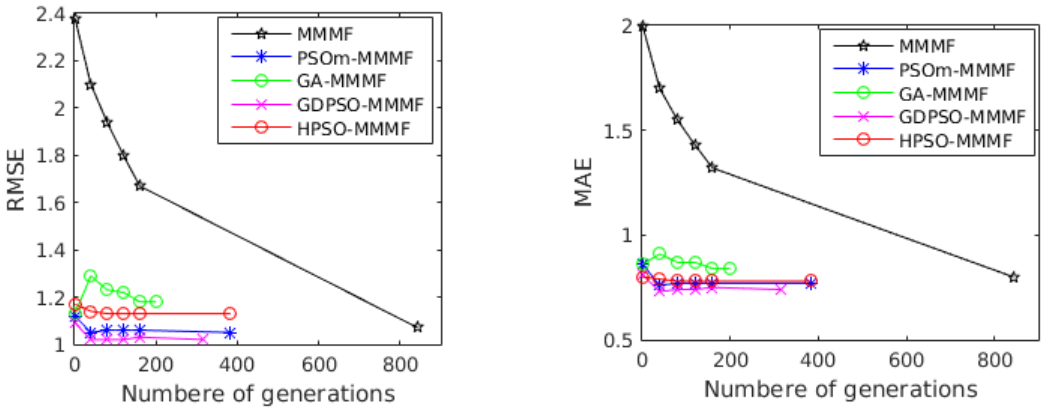


Figure 3.6: RMSE and MAE for MMMF, GDPSO-MMMF, PSOm-MMMF, HPSO-MMMF and GA-MMMF for 10k dataset

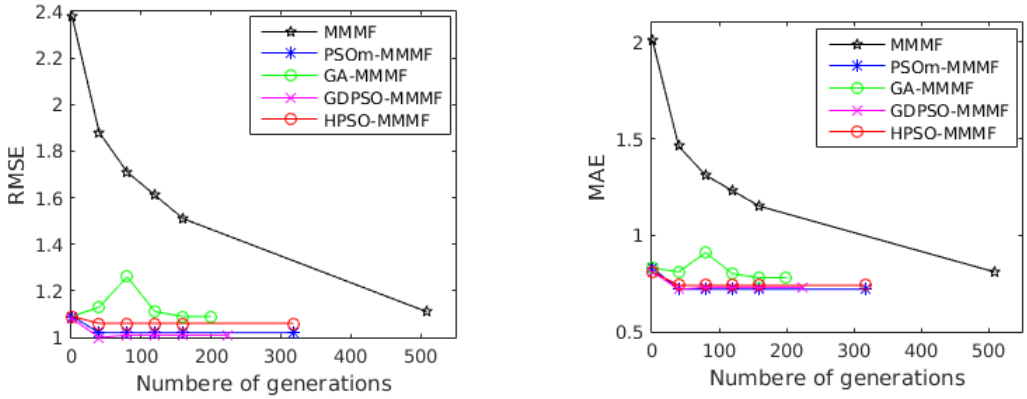


Figure 3.7: RMSE and MAE for MMMF, GDPSO-MMMF, PSOm-MMMF, HPSO-MMMF and GA-MMMF for 50k dataset

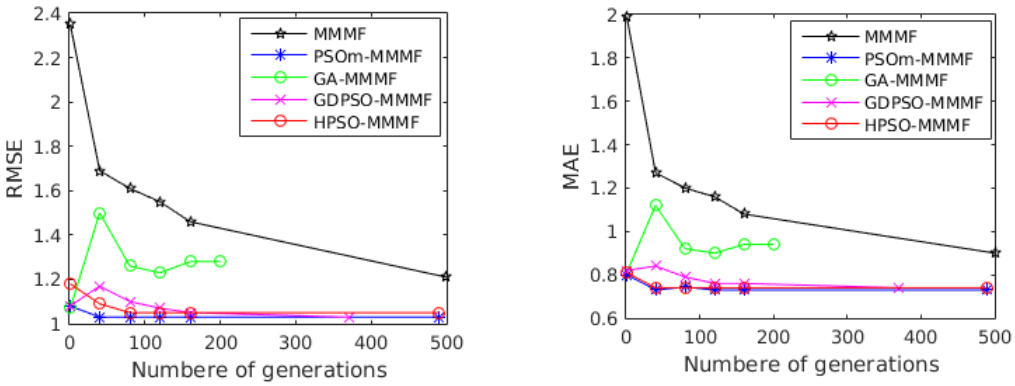


Figure 3.8: RMSE and MAE for MMMF, GDPSO-MMMF, PSOm-MMMF, HPSO-MMMF and GA-MMMF for 100k dataset

Size	MMMF	GDPSO-MMMF	PSOm-MMMF	HPSO-MMMF	GA-MMMF	PSO-MMMF [19]
10k	1.82	1.033	1.06	1.13	1.21	1.63
50k	1.7	1.02	1.03	1.06	1.14	1.53
100k	1.645	1.08	1.03	1.05	1.27	1.34

Table 3.1: Comparison of the proposed algorithms with MMMF and [19] in terms of RMSE for different dataset sizes.

Size	MMMF	GDPSO-MMMF	PSOm-MMMF	HPSO-MMMF	GA-MMMF	PSO-MMMF [19]
10k	1.07	0.75	0.78	0.78	0.87	1.25
50k	1.32	0.74	0.73	0.74	0.82	1.12
100k	1.266	0.785	0.751	0.74	0.94	0.95

Table 3.2: Comparison of the proposed algorithms with MMMF and [19] in terms of RMSE for different dataset sizes.

3.3 Summary

We outlined a framework for combining metaheuristic techniques like particle swarm optimisation and genetic algorithm with matrix factorisation based collaborative filter-

ing methods. We have demonstrated how this combination can improve the accuracy of a recommender system in terms of RMSE and MAE. Our main objective was to employ different variants of heuristic based optimization techniques with maximum margin matrix factorisation and observe the corresponding improvements. The improvement of heuristic based MMMF models is due to the fact that promising solutions are achieved in lesser iterations. This is because of the characteristics of the particle swarm optimization approach wherein we have several possible solutions scattered in the search space to start with. The potential for searching optimal solutions at different regions parallelly is also there and exchanging information with each other makes it possible to guide themselves to the optimal solution efficiently in lesser iterations. Similarly, genetic algorithm also maintains several possible solutions scattered in search space wherein only the fittest chromosomes reproduce and evolve and thereby the potential of getting promising solutions faster is possible. When the exploration capability of population based search algorithms is combined with gradient search direction of MMMF, the proposed models are able to achieve accurate prediction of missing ratings. We have shown improvised variants of PSO based MMMF that outperform the base PSO and the base MMMF. Though GA based MMMF have not shown the most promising results, it improves the base MMMF by influencing the gradient search of MMMF. We are convinced from our extensive experiments that a combined approach of metaheuristics with matrix factorisation models helps in yielding better accuracy in recommender systems.

Chapter 4

A Two-Step Process of Discovering User-Item Subgroups Using Metaheuristics in Neighbourhood Method

In the previous chapter we have seen how matrix factorisation methods like maximum margin matrix factorisation can be combined with metaheuristic techniques like particle swarm optimisation and genetic algorithm. Our experimental results revealed that the prosed methods are better in terms of RMSE and MAE. Unlike matrix factorization, neighbourhood based collaborative filtering exploits similarity of a group of users/items while predicting the missing ratings. The neighbourhood methods rely on interactions between users, or alternatively between items. The fundamental assumption in neighbourhood method is that once certain users with similar preferences are found based on some similarity in the items they like, then they will have similar tastes on all the remaining items. However, this concept is not entirely tenable as it is not reasonable to consider that two users will have similar tastes on the entire set of items because of having similarity between only a subset of items [68]. Using such information in prediction computation tend to degrade the performance of the algorithms. Moreover, one's preferences generally revolve around some topics rather than on every available topics and it is more reasonable to assume that a group of users are more relevant on a subset of items. In this chapter, we focus on discovering highly correlated

set of users based on a subset of items rather than using the entire information of similar users in prediction computation. Identification of such subsets of information from massive data sets is a popular approach in estimating missing values of the data matrix in domains such as bioiformatics [15]. We call the group of users that has similar tastes on a subset of items as *correlated user-item subgroups*.

In this chapter, we propose algorithms that discover the correlated user-item subgroups and employ the discovered information in the prediction of missing ratings by using the *least squares* approach. Least squares (LS) approach [13, 15, 28, 39, 63] have been successfully applied to estimate missing values of data matrix as can be seen from the literature. Our objective is to discover highly correlated user-item subgroups in neighbourhood based collaborative filtering and predict the missing ratings by using the rating information of the subgroups with the least square method. In this chapter, we propose four algorithms that are centered on discovering the most correlated user-item subgroups by making use of different techniques. All the four proposed algorithms have a common structure of three steps in which the first step finds a group of similar users for the target user, the second step discovers a subset of similar items based on the preferences of the selected similar group of users, and the last step is the prediction computation. The four proposed algorithms are different in the second step that discovers the subset of similar items. The proposed algorithms explore different techniques such as 1) using a threshold to check the correlated degree of the subgroups, 2) leverage metaheuristics approach like GA in searching for highly correlated user-item subgroups, 3) flexible number of similar users for each target user based on their similarity degree and 4) a hybrid of the above.

The Sections and subsections of this chapter are organized as follows. The least squares imputation approach for collaborative filtering is discussed in Section 4.1. In Section 4.2 we outline the four algorithms that has been proposed. This Section has been divided into four subsections wherein subsection 4.2.1 describes the first algorithm that is threshold based and subsection 4.2.2 presents the second algorithm which is a metaheuristic based collaborative filtering algorithm. The third subsection 4.2.3 presents the third proposed algorithm which is iterative based, and the fourth algorithm that is a hybrid of iterative and metaheuristic based algorithm is presented in the fourth subsection. 4.2.4. Experimental results related to all the four proposed algorithms are outlined in Section 4.3 and with Section 4.4 we conclude the chapter.

4.1 Background: Local Least Squares Imputation of Collaborative Filtering

In the literature, several researchers have employed Local Least Squares Imputation framework in prediction of missing entries of a data matrix [13, 15, 28, 39]. In this section, we formulate the least squares approach for prediction in order to suit the traditional collaborative filtering method. The prediction of the missing entries in a partially observed user-item rating matrix can be performed by identifying the local correlation of the user-item matrix. Therefore, the framework is named as *Local Least Squares imputation of Collaborative Filtering, (LLSimCF)*. The missing entries of the user-item rating matrix are initially filled with row-wise averaged value to simplify the problem. The fundamental steps of LLSimCF framework to predict the unknown ratings of the data matrix are given below.

1. Discover similar users, $u_{S_i}^T \in R^{1\times n}, 1 \leq i \leq k$.
2. Prediction of missing ratings by using least squares method

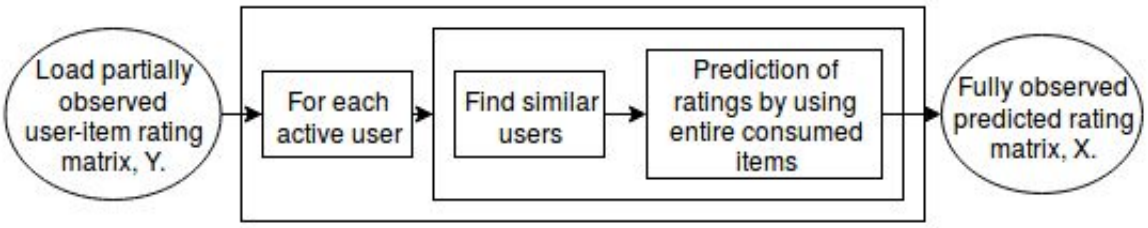


Figure 4.1: Block diagram of LLSimCF

Figure (4.1) shows the block diagram of LLSimCF.

4.1.1 Discover similar users

The first step in typical neighbourhood method is to discover a group of similar users for an active user by considering the entire consumed items of the active user. So, we first discover k most similar users for each active user by identifying mutual preferences of the users. Different similarity measures have been used, such as Pearson

correlation, adjusted cosine similarity and Euclidean distance. The definitions of the similarity measures are explained in Section 2.2.2 of Chapter 2. The selected k most similar users of an active user are represented by $u_{S_i}^T \in R^{1\times n}, 1 \leq i \leq k$.

4.1.2 Prediction of missing ratings by using least squares method

In prediction computation, the rating information obtained from the similar users $u_{S_i}^T$ are used with Least Squares (LS) approach in order to predict missing ratings for the active user. To predict p missing ratings with the LS approach, we generate two matrices $A \in R^{k\times(m-p)}$ and $B \in R^{k\times p}$ and a vector $w \in R^{(m-p)\times 1}$ based on the k most similar users [39]. The matrix B is constructed by using ratings of the similar users $u_{S_i}^T$ for the items unknown to the active user. The matrix A is constructed by using ratings of the similar users $u_{S_i}^T$ for the consumed items of the active user. The vector w consists of known $(m-p)$ ratings of the active user. A vector a of an active user represents the missing ratings of the items, such that $a = (\alpha_1, \alpha_2, \cdots, \alpha_p)^T$. Hence, the rating information of an active user u_t is represented by eq. (4.1).

$$u_t^T = [a^T \;\; w^T] \tag{4.1}$$

Similarly, the rating information of k similar users is given by eq. (4.2).

$$\begin{pmatrix} u_{S_1}^T \\ \vdots \\ u_{S_k}^T \end{pmatrix} = [B \;\; A] \tag{4.2}$$

Now, we represent the rating information of the active user and k most similar users by using Eqn. (4.3).

$$\begin{pmatrix} u_t^T \\ u_{S_1}^T \\ \cdots \\ u_{S_k}^T \end{pmatrix} = \begin{pmatrix} a & w^T \\ B & A \end{pmatrix} \tag{4.3}$$

In order to predict missing ratings of the active user, the least square problem is formulated as in Eqn. (4.4) where x is a vector with the coefficients of linear combinations.

$$\min_x ||A^T x - w||_2 \tag{4.4}$$

The missing ratings of the active user can be predicted by using rating information of the k most similar users and the coefficients of linear combination found by LS formulation [39]. Thus, the missing ratings represented in vector $a = (\alpha_1, \alpha_2, \cdots, \alpha_p)^T$ can be predicted by using Eqn. (4.5) which is derived from LS formulation, where $(A^T)^\dagger$ represents the pseudoinverse of A^T.

$$a = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_p \end{pmatrix} = B^T x = B^T (A^T)^\dagger w \tag{4.5}$$

Let us take an example with two unknown ratings α_1 and α_2 and represent the example using Eqn. (4.3). Let the number of consumed items be denoted by $consumed$ where $consumed = m - 2$.

$$\begin{pmatrix} u_t^T \\ u_{S_1}^T \\ \cdots \\ u_{S_k}^T \end{pmatrix} = \begin{pmatrix} \alpha_1 & \alpha_2 & w_1 & w_2 & w_3 & w_{consumed} \\ B_{1,1} & B_{1,2} & A_{1,1} & A_{1,2} & A_{1,3} & A_{1,consumed} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ B_{k,1} & B_{k,2} & A_{k,1} & A_{k,2} & A_{k,3} & A_{k,consumed} \end{pmatrix} \tag{4.6}$$

The unknown ratings α_1 and α_2 can be predicted by using matrix B and the coefficients of linear combinations x. $u_{S_1}^T$ to $u_{S_k}^T$ are k similar users of the active user u_1. α_1 and α_2 can then be given as

$$\alpha_1 = B_{1,1}x_1 + B_{2,1}x_2 + \cdots + B_{k,1}x_k$$
$$\alpha_2 = B_{1,2}x_1 + B_{2,2}x_2 + \cdots + B_{k,2}x_k$$

The local least square imputation framework, ***LLSimCF*** gets the advantage from the rating information of similar users in the optimization step of the LS approach. Let us take an example to elaborate the prediction formulation of the least square method. Consider a partially observed user-item rating matrix $Y \in R^{4\times 7}$ such that the number of users and items are 4 and 7 respectively and the users and items are represented in rows and columns respectively. Each entry cell of the matrix corresponds to the rating of the respective user given to the corresponding item.

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} 5 & 3 & 3 & 3 & 0 & 0 & 3 \\ 4 & 3 & 0 & 0 & 5 & 1 & 4 \\ 0 & 4 & 1 & 1 & 4 & 2 & 0 \\ 5 & 4 & 4 & 0 & 1 & 1 & 3 \end{pmatrix}$$

For instance, let the active user be u_1 and the number of similar users k be 2. The two most similar users of the active user are u_2 and u_4. Initially, we fill the missing entries of the matrix with the row-average value, and then the rating matrix is as given below:

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} 5 & 3 & 3 & 3 & 3.4 & 3.4 & 3 \\ 4 & 3 & 3.4 & 3.4 & 5 & 1 & 4 \\ 2.4 & 4 & 1 & 1 & 4 & 2 & 2.4 \\ 5 & 4 & 4 & 3 & 1 & 1 & 3 \end{pmatrix}$$

Note that the active user has two missing ratings in the original rating matrix. Thus, the two missing ratings, $a = (\alpha_1, \alpha_2)^T$, are represented by a vector a. Now, we represent the rating information of both the active user and the similar users by using least squares formulation which is given by Eqn. (4.6):

$$\begin{pmatrix} u_1^T \\ u_2^T \\ u_4^T \end{pmatrix} = \begin{pmatrix} \alpha_1 & \alpha_2 & 5 & 3 & 3 & 3 & 3 \\ 5 & 1 & 4 & 3 & 3.4 & 3.4 & 4 \\ 1 & 1 & 5 & 4 & 4 & 3 & 3 \end{pmatrix}$$

From eq. (4.4), we obtained coefficients of linear combinations such as $x_1 = 0.2941$ and $x_2 = 0.6275$. Thus, the missing ratings α_1 and α_2 can be imputed by computing the following

$$\alpha_1 = B_{1,1}x_1 + B_{2,1}x_2 = 5 \times 0.2941 + 1 \times 0.6275 = 2.098$$
$$\alpha_2 = B_{1,2}x_1 + B_{2,2}x_2 = 1 \times 0.2941 + 1 \times 0.6275 = 0.9216$$

Thus, α_1 and α_2 are predicted as 2 and 1 respectively.

4.2 Proposed Algorithms

The framework *LLSimCF* described in the previous subsection predicts unknown ratings of user-item rating matrix by using rating information of similar users. Note that the prediction procedure in LLSimCF considers entire consumed items which may involve irrelevant items with reference to the corresponding active user, thereby resulting in a possibility of obtaining less accurate prediction. To solve this problem, we propose a novel framework that discovers a highly correlated user-item subgroup for each

active user and then the rating information of the discovered subgroup is used in the prediction by using the least square method. We propose four algorithms related to our proposed framework which helps in predicting the missing ratings by considering only highly correlated user-item subgroups. The subgroup is constructed by making use of a *Two-Step Process* as given below:

(1) A set of similar users are selected through a similarity measure that considers the entire consumed items of the active user.

(2) The proposed algorithm discovers a subset of highly correlated items based on the subset of users from step 1.

The discovered subset of users and items construct a subgroup which has useful information for a possible accurate prediction of the missing ratings. The users and items in the subgroups may belong to multiple subgroups. The generalized block diagram of the four proposed algorithms is shown in Fig. (4.2) and the differences of the four proposed algorithms are highlighted in the inner block such that the construction of correlated user-item subgroup is presented. The basic steps of the proposed algorithms are:

1. Discovery of a set of similar users by using similarity metric.

2. Discovery of a subset of similar items from the consumed items of the selected similar users by using different techniques according to each proposed algorithm. Thus, correlated user-item subgroups are constructed.

3. Prediction of the missing ratings by using the least square framework based on the discovered correlated user-item subgroup.

Unlike LLSimCF framework, all the four proposed algorithms follow three basic steps to compute the prediction. Step 1 and 3 are similar to the explanation of step 1 and 2 of LLSimCF framework in the previous section except that the least squares method will be applied on only the correlated subgroups. Our proposed algorithms make use of the step 2 which is explained in subsections 4.2.1 to 4.2.4, namely *SLLSimCF*, *GA-SLLSimCF*, *ISLLSimCF*, and *I-GA-ISLLSimCF*. The block diagram of the proposed framework is shown in Fig.(4.2). The block diagram clearly shows the additional block of the proposed framework as compared to the underlying LLSimCF algorithm. The

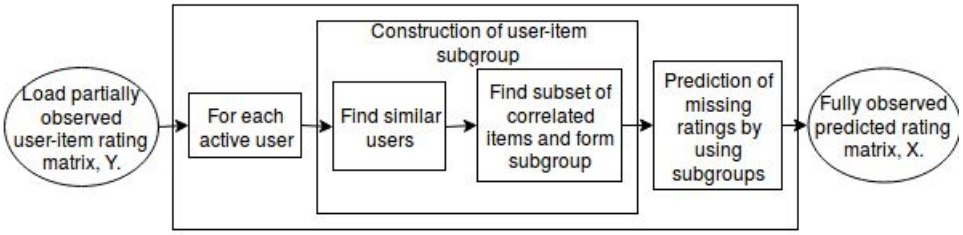


Figure 4.2: Block diagram of the proposed method

addition is the block related to the construction of user-item subgroup which is the goal of the proposed framework.

4.2.1 Subgroup based Local Least Squares Imputation of Collaborative Filtering (SLLSimCF)

This proposed algorithm discovers correlated user-item subgroup by using a threshold that determines the accepted degree of correlation of the user-item in the similar user's profile. The proposed algorithm is termed as Subgroup based Local Least Square imputation of CF algorithm, (*SLLSimCF*). Once the correlated user-item subgroups are discovered, the proposed algorithm predicts the missing ratings by using the least squares method with the information of the discovered user-item subgroups. Note that the proposed algorithm considers only highly correlated user-item subgroups in prediction computation in order to achieve higher accuracy as compared to the underlying algorithm that considers the entire consumed items including irrelevant information. As mentioned earlier, initially the set of k similar users of the subgroup is discovered by using popular similarity measures and then the subset of similar items based on the selected k similar users is discovered by using the following technique. A new matrix R is generated that consists of rating information of the k similar users[15]. The matrix R is shown by Eqn. (4.7)

$$R = B^T A \tag{4.7}$$

To select a subset of highly correlated items from the consumed items of the active user with respect to the missing entry at j^{th} column, the uncorrelated items from the k similar users are removed by using eq. (4.8). The rating information R of k similar users is compared with the threshold, θ, to determine the degree of correlation of the

subset of items.

$$\begin{aligned} \tau &= (r_{j,max} + r_{j,min})/2 \\ \theta &= \tau - (\tau/3) \end{aligned} \tag{4.8}$$

where $r_{j,max}$ and $r_{j,min}$ correspond to maximum and minimum value in row j. The user j with respect to item v is classified as *correlated* if $r_{j,v} \geq \theta$ where $r_{j,v}$ denotes (j, v) entry of the R matrix. The minimum correlation allowed is the threshold θ. The items with higher correlation are the ones to be discovered and is our prime focus whereas the items with lower correlation are considered as irrelevant items and removed. Through this process, a set of highly correlated items are discovered. The active user with correlated items are represented by Eqn. (4.9)

$$u_t^T = [\alpha_j \;\; w_j^T] \tag{4.9}$$

where α_j represent j^{th} missing rating of the active user and w_j^T represents known ratings of the active user u_t^T for the items that are correlated with the j^{th} missing column. The ratings of the subset of selected items for the k similar users is represented by Eqn. (4.10)

$$\begin{pmatrix} u_{S_1}^T \\ \vdots \\ u_{S_k}^T \end{pmatrix} = [b_j \;\; A_j] \tag{4.10}$$

where b_j represents ratings of k similar users for the j^{th} missing column and A_j denotes the ratings of the selected subset of correlated items of the k similar users. The least squares formulation with respect to user-item subgroup is given by Eqn (4.11).

$$\min_x ||A_j^T x - w_j||_2 \tag{4.11}$$

Therefore, j^{th} missing rating can be predicted as

$$\alpha_j = b_j x = b_j (A_j^T)^\dagger w_j \tag{4.12}$$

Now, the missing ratings of the active user can be predicted as explained in the above example. The proposed algorithm is described in algorithm (5). The proposed algorithm (5) terminates when all the missing ratings are imputed for every user by using LS framework.

Algorithm 5: SLLSimCF

Input: Matrix Y, partially observed rating matrix, k, T_0
Output: Matrix X, fully observed predicted rating matrix
Find k nearest users for each active user by using a similarity measure;
for *each active user* **do**
    Create two matrices A and B and a vector w;
    Perform $R = B^T A$;
    for *each missing entry* j **do**
        Identify correlated subset of items by using eq.(4.8);
        Create new matrix A_j according to the selected subset of items;
        Create vector b_j;
        Apply LS method by using b_j, w_j and the new A_j to predict the missing ratings;
Finally, a fully observed predicted rating matrix X is obtained.

4.2.2 Genetic Algorithm oriented Subgroup based Local Least Squares Imputation of Collaborative Filtering (GA-SLLSimCF)

Here, we propose a metaheuristic based algorithm, named as *GA-SLLSimCF* (Genetic Algorithm oriented Subgroup based Local LS imputation of CF), that discovers highly correlated user-item subgroup through genetic algorithm for each active user. The least squares method is used with the retrieved subgroups to effectively predict the missing ratings. As mentioned above, the proposed algorithm, GA-SLLSimCF, first finds k similar users by using a similarity metric. Then, a subset of highly correlated items based on the set of k similar users is discovered by using GA that constructed the desired user-item subgroup. The proposed algorithm applies LS framework with the discovered user-item subgroup in order to predict missing ratings of the rating matrix. The user-item subgroup can be considered as a sub-matrix where rows represent users and columns represent items. The degree of correlation of the items in the subgroup is determined by mean squared residue score. Note that the word subgroup and sub-matrix are used interchangeably as they represent the same meaning. The characteristics of GA regarding the proposed algorithm is explained in the following subsections.

4.2.2.1 Encoding of chromosome

The candidate solutions of the population are also termed as chromosome. Binary encoding is used to encode the chromosome such that the length of the chromosome is

denoted by len in which len is the number of consumed items of the active user. The bit 1 in the chromosome represents selection of the corresponding consumed item whereas bit 0 represents that the item is irrelevant. Each chromosome represents a subset of correlated items for an active user. Let's take the example from earlier subsection 4.1.2, namely *Prediction of missing ratings by using least squares method*, in which the user-item rating matrix $Y \in R^{4\times7}$ is considered. Here, the number of consumed items, len, is 5. A new matrix is constructed such that rating information of the remaining users for the consumed items of the active user are considered. The new matrix is shown below.

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} 5 & 3 & 3 & 3 & 3 \\ 4 & 3 & 3.5 & 3.5 & 4 \\ 2.4 & 4 & 1 & 1 & 2.4 \\ 5 & 4 & 4 & 3 & 3 \end{pmatrix}$$

If a chromosome is encoded as 00111, it means that three consumed items, i_3, i_4, i_5, are selected.

0	0	1	1	1

The sub-matrix associated with this chromosome is shown below and is highlighted in bold letters in which u_1 is the active user and u_2, u_4 are the similar users. Similarly, there are several sub-matrices in a population associated with each chromosome.

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} 5 & 3 & 3 & 3 & 3 \\ 4 & 3 & \mathbf{3.5} & \mathbf{3.5} & \mathbf{4} \\ 2.4 & 4 & 1 & 1 & 2.4 \\ 5 & 4 & \mathbf{4} & \mathbf{3} & \mathbf{3} \end{pmatrix}$$

4.2.2.2 Quality of chromosome

At this stage it stands to reason, how the quality of a chromosome discovered by GA can be evaluated by using the two measures [10] given below.

1. *Mean squared residue:*

   Mean squared residue evaluates the degree of correlation of an item with the

remaining items in a subgroup. The value of the residue and correlation rate of the item are inversely proportional. The smaller value of residue shows a high correlation of the items, hence better quality of the user-item subgroup. The quality of subgroup can be defined by the mean squared residue of all the items in the subgroup. The mean squared residue of subgroup $S \times T$ is evaluated as

$$msr_{ST} = \frac{1}{|S|.|T|} \sum_{s \in S, t \in T} (x_{st} - x_{sT} - x_{St} + x_{ST}) \tag{4.13}$$

In the Eqn. (4.13), x_{sT} corresponds to mean row, x_{St} corresponds to mean column and x_{ST} corresponds to mean of all entries in the subgroup.

2. *Row variance:*
   The row variance guarantees that the subgroup allows users to express diverse preferences yet correlated across some set of items. It is defined by eq. (4.14).

$$variance_{ST} = \frac{\sum_{s \in S, t \in T} (x_{st} - x_{sT})^2}{|S|.|T|} \tag{4.14}$$

4.2.2.3 Fitness function

The quality of a chromosome is determined by evaluating the fitness function of the chromosome which is the objective function of the problem. The fitness function is a minimization function, and is given by Eqn. (4.15). The fitness value determines the degree of correlation of the user-item subgroup. The lesser the fitness value, the better the quality of a subgroup is. A subgroup having a good quality means that the correlation among the users on the set of selected items is strong.

$$f(C) = \frac{msr(C)}{\delta} + \frac{1}{variance(C)} \tag{4.15}$$

where $msr(C)$ and $variance(C)$ represent mean squared residue and row variance of chromosome C respectively. The δ is a user-defined upper boundary of the acceptable dissimilarity of the subgroup. Since the fitness function is a minimization function, mean squared residue is expected to be a small value as it is in the numerator and the row variance to be a large value as it is in the denominator so as to make the selected

subgroup a good candidate by having a small fitness value. Row variance of a subgroup makes sure that the preferences of the users fluctuate but is correlated across a subset of items. Note that the best subgroup in the population has the smallest fitness value.

4.2.2.4 Genetic Operators

Genetic operators in GA consists of three main operators, namely selection, crossover and mutation operators. In selection operator, tournament selection is used in which default size of the number of the candidate solution in a tournament is set to two. It means that two chromosomes are randomly selected and their fitness values are compared among each other. The chromosome with the smaller fitness value is selected as parent, and kept in mating pool to generate offspring for the next generation. Selection process is repeated several times until the mating pool is fully-filled. After the selection process, the selected chromosomes from the mating pool participates in the recombination process which are crossover and mutation operator.

In crossover operation, one-point crossover method is employed to recombine the parents and form offsprings. In mutation operation, bit-flip mutation process is employed to get a diverse population in the search of nearly optimal solutions. Finally, fitter chromosomes from the mating pool and original population are selected to evolve in the successive generations. The elitism concept is adopted to make sure that some percentage of good chromosomes are passed on to the next generation with the assumption that current population has some good characteristics to descend to future generations. The parameter values are given in section 4.3.2 namely *Parameters Setting*. The proposed algorithm is given in algorithm (6). Note that the proposed algorithm employs GA for each active user in searching correlated subset of items. The GA part of the proposed algorithm terminates when the user-defined parameter, $max - iteration$, is satisfied. Finally, the proposed algorithm terminates when all the missing ratings of the users are predicted.

4.2.3 Iterative Subgroup based Local Least Squares Imputation of Collaborative Filtering (ISLLSimCF)

In this proposed method, we tune the algorithm to behave in an iterative manner by selecting different k similar users for each user and passing the current predicted ma-

Algorithm 6: GA-SLLSimCF

Input: Matrix Y, partially observed rating matrix.
Output: Matrix X, fully observed predicted rating matrix
Find k similar users for each active user;
for *each active user* **do**
    Create matrices A, B and a vector w;
    Apply GA to select highly correlated subgroup as follows;
        Initialize population randomly;
        Evaluate fitness function of each chromosome by using Eqn. (4.15);
    while *(max-iteration is not met)* **do**
            Select parents;
            Produce offspring by using crossover and mutation operator;
            Select the fittest chromosomes for next generation;
        Return the best chromosome when the stopping criteria is met;
    for *each missing entry* j **do**
        Create new matrix A_j according to the selected subset of items;
        Create vector b_j;
        Apply LS method by using A_j, b_j and w_j to predict the missing ratings;
Finally, a fully observed predicted rating matrix X is obtained.

trix as input to the successive iterations until the algorithm converges to an optimal solution. The algorithm is termed as Iterative Subgroup based Local Least Square imputation of CF *ISLLSimCF*). The aim of the iterative procedure is to enhance the selection of k similar users, and thereby enhance the prediction of missing ratings [13]. The assumption of the proposed algorithm to select different number of similar users for each active user is based on the reasoning that each active user need not have same number of similar users as each user has different number of ratings available and is more reasonable to relax the restriction. However, the proposed algorithm has a threshold, $thres$, that is used to measure acceptable distance between two users to filter dissimilar users. The value of the threshold, $thres$, is set by using known ratings in the user-item rating matrix, which is the *mean* of all the distances between the active users and the other users. If the distance between the active user and another user is less than the threshold $thres$, then the other user is selected as a similar user. The threshold $thres$ is defined by Eqn. (4.16) where $ratioDistance$ is a parameter. We carried out experiments with different parameter values ranging from 0.2 to 0.8 to fix the value of

$ratioDistance$ and empirically selected the value which gave the most accurate result.

$$thres = meanDistance \times ratioDistance \tag{4.16}$$

To select a highly correlated subset of items for a set of users, the iterative based proposed algorithm employs the same threshold, θ, based technique from algorithm (5) which is described by Eqn. (4.8). Once the user-item subgroup is constructed, the proposed algorithm predicts the missing ratings by using LS framework with the rating information of the discovered user-item subgroup. The proposed algorithm is

Algorithm 7: ISLLSimCF

Input: Matrix Y, partially observed rating matrix, k, T_0
Output: Matrix X, fully observed predicted rating matrix
while *(max-iteration is not met)* **do**
  for *each active user* **do**
    Find k similar users by using a distance threshold $thres$;
    Create two matrices A and B and a vector w;
    Perform $R = B^T A$;
    for *each missing entry* j **do**
      Identify correlated subset of items by using Eqn. (4.8);
      Create new matrix A_j according to the selected subset of items;
      Create vector b_j;
      Apply LS method by using b_j, w_j and the new A_j to predict the missing ratings;
  Imputed rating matrix is passed as input matrix to next iteration.
Finally, a fully observed predicted rating matrix X is obtained.

described in algorithm (7). Note that the value of k in the proposed algorithm varies in each iteration.

4.2.4 Iterative based Genetic Algorithm oriented Subgroup based Local Least Squares Imputation of Collaborative Filtering (I-GA-SLLSimCF)

In this subsection, we propose an algorithm that is a hybrid of the iterative (*ISLL-SimCF*) and metaheuristic based (*GA-SLLSimCF*) algorithms which were discussed in the previous subsections. The advantages of both the algorithms are merged together

in this proposed framework which is termed as Iterative based Genetic Algorithm oriented Subgroup based Local Least Square imputation of CF (*I-GA-SLLSimCF*). The proposed algorithm *I-GA-SLLSimCF* takes advantage of the relaxation in discovering different number of similar users for each active user from iterative based algorithm (*ISLLSimCF*), and the exploration capacity of GA in identifying a correlated subset of items from the GA based algorithm (*GA-SLLSimCF*). By using the combined framework of the iterative and metaheuristic searching algorithm, subsets of highly correlated items are discovered which is further utilized in the prediction computation of missing ratings by using LS framework. The fully observed predicted rating matrix is further passed to the next iteration as input and the procedure is repeated for multiple times until it is converged to a nearly optimal solution or termination condition is satisfied. The proposed algorithm is described in algorithm (8).

Algorithm 8: GA-ISLLSimCF

Input: Matrix Y, partially observed rating matrix.
Output: Matrix X, fully observed predicted rating matrix
while *(max-iteration is not met)* **do**
  for *each active user* **do**
    Find k similar users by using a distance threshold $thres$;
    Create matrices A, B and a vector w;
    Perform $R = B^T A$;
    Apply GA to select submatrix of highly correlated items as follows;
      Initialize population randomly;
      Evaluate fitness function of each chromosome by using equation(4.15);
    while *(ga-max-iteration is not met)* **do**
      Select parents;
      Produce offspring by using crossover and mutation operator;
      Select the fittest chromosomes for next generation;
      Return the best chromosome when the max-iteration is met;
    for *each missing entry* j **do**
      Create new matrix A_j according to the selected subset of items;
      Create vector b_j;
      Apply LS method by using A_j, b_j and w_j to predict the missing ratings;
  Imputed rating matrix is passed as input matrix to next iteration.
Finally, a fully observed predicted rating matrix X is obtained.

4.3 Empirical Analysis of User-Item Subgroups Algorithms via Two-Step Process for Prediction and Recommendation

4.3.1 Experimental Settings

We carried out experiments with all the proposed algorithms by using a benchmark dataset *Movielens* that consists of 100k number of ratings. The proposed algorithms are also experimented with smaller datasets, namely 50k ratings and 10k ratings which are extracted from the Movielens 100k dataset. The dataset contains ratings in the range of 1 to 5. We used 1M movielens dataset as well for a part of the experiments. In both the datasets, we separated the dataset into 20% test set and 80% training set randomly, in such a way that 20% known ratings of each user is randomly selected and marked as unknown by replacing the true ratings with zero. The new dataset with test ratings filled as zero is the training dataset. To show the robustness of the proposed algorithms, we have measured the algorithms for prediction and recommendation tasks. For prediction evaluation, we have employed the well known accuracy metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). In recommendation evaluation, we have used top N = 10 as it is not desirable to suggest a long list of items but a shorter and possibly interesting items. We have employed the popular metrics for recommendation such as precision, recall, F1 score and mean average precision.

4.3.2 Parameter Setting

To identify a suitable parameter value, several preliminary experiments are performed such that each proposed model is executed multiple times with different values of parameters so as to select a suitable value that converges the algorithm to an optimal solution. The parameter analysis could be a prolonged operation as there are uncountable combination of values. The parameter values are fixed in such a way that the most accurate result among a set of random values is selected. The same parameter values are used in all the proposed algorithms for a fair evaluation. We used 100k Movielens dataset to experiment different parameter values.

- Number of similar users (k):
  Figure (4.3c) shows the parameter tuning of the number of similar users k. When the value of k increases, we observe less accurate result of the predicted rating matrix which is shown in the figure. Moreover, if the number of similar users is too small, it restricts the possibility to discover more preferences of an active user to a small group of users. Hence, we set k to 10 for all the algorithms except the iterative algorithm since the value of k varies for each user in the case of the iterative algorithm.

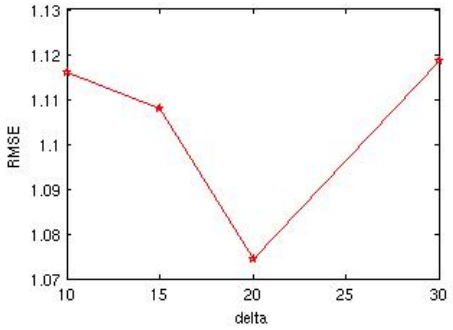
- Parameter specific for the models *I-SLLSimCF* and *I-GA-SLLSimCF*, $ratioDistance$:
  The graph shown in figure (4.3d) suggests that the performance of the iterative based algorithms degrade as we increase the size of the parameter $ratioDistance$. The larger the value of the parameter, the algorithm selects larger number of similar users and this leads to a decrease in performance as we can observe from the graph (4.3c). The parameter value is set to 0.1. The termination condition of both the algorithms is set when user-defined maximum iteration is met or when the predicted rating matrix from current iteration is the same as the re-imputed rating matrix of the previous iteration.

- Population size and termination condition of Genetic Algorithm:
  The ideal size of the population of GA is selected as 50. The termination condition of GA, $ga - max - iteration$, is set to 50.

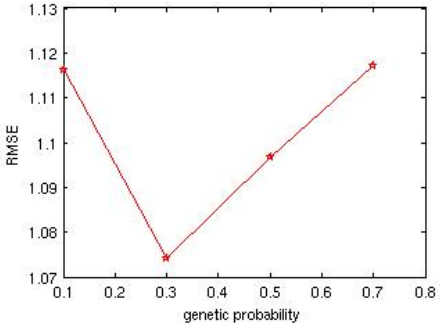
- Probability of genetic operators:
  The performance of the proposed algorithm GA-SLLSimCF according to different values of the parameter that monitor the genetic probability is shown in figure (4.3b). The genetic operators of GA such as crossover and mutation are performed according to the probability value which is fixed at 0.3 since it has shown the best performance. The larger portion of the probability allows the proposed algorithms to perform crossover whereas the smaller portion allows to perform mutation since more mutation would make the refined search space unnecessarily larger. Moreover, elite chromosomes are selected in each generation at the percentage of 30 in order to retain the elite chromosomes in successive

generations. The parameter δ for the fitness function is set to 20 after experiments on different values of the parameter which is shown in figure (4.3a).

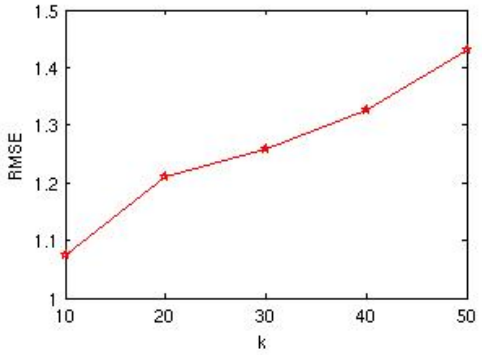
- The number of subgroup in co-clustering algorithm is set to 15.



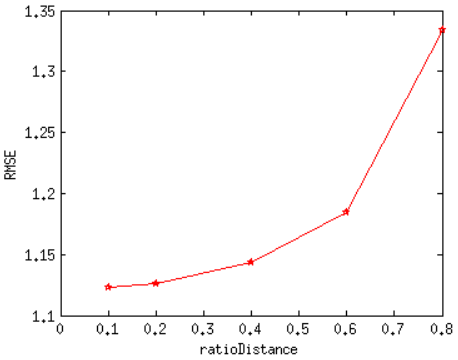
(a) Different values of δ in GA-SLLSimCF



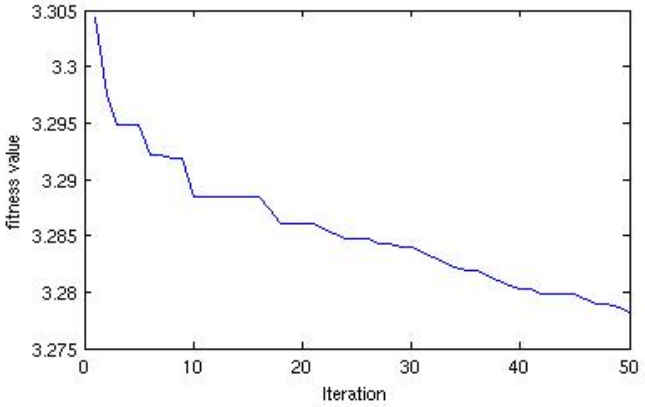
(b) Different values of ρ in GA-SLLSimCF



(c) Different values of k in GA-SLLSimCF



(d) Different values of $ratioDistance$ in ISLLSimCF



(e) Graph showing fitness value in every generation in GA-SLLSimCF

Figure 4.3: Graphs show parameter tuning results of several parameters used in the proposed algorithms *GA-SLLSimCF*, *ISLLSimCF* and *SLLSimCF*

4.3.3 Comparison

For comparative analysis, we employed well known state-of-the-art collaborative filtering algorithms, namely item based neighbourhood algorithm, clustering based algorithm and a latent factor model. To our knowledge, there is no extensive research based on user-item subgroup based CF algorithm that excludes irrelevant items in the computation of prediction. We compared our proposed algorithm with the following state-of-the-art algorithms.

- *Item Based CF (IB)*:
  Item based CF [55] is one of the well known traditional CF algorithms for prediction of ratings which considers rating information of all the consumed items of the similar users regardless of their dissimilarity among the items. We use *IB* to justify the prediction quality of the proposed subgroup based algorithm. The adjusted cosine similarity is used to find similarity among users.

- *Maximum Margin Matrix Factorization (MMMF)*:
  MMMF is one of the most popular model based CF algorithm [52]. MMMF is used in our comparison as the base algorithm for collaborative filtering so as to justify the significance of user-item subgroups in prediction.

- *Particle Swarm Optimization based MMMF (PSO-MMMF)*:
  This approach showed a new direction to improve the gradient search of MMMF by incorporating a component of Particle Swarm Optimization [19]. The approach is considered to compare the accuracy in terms of prediction of missing ratings.

- *Multiclass Co-Clustering (MCoC)*:
  This is one of the most popular user-item subgroup based CF that considers only correlated subgroups in the prediction of top N recommendation items. The MCoC used a soft clustering technique that allows multiple class cases [68]. In addition, Single-class Co-Clustering (SCoC) is experimented in which subgroups in the context of SCoC are disjoint. The difference between these subgroup based algorithms and our proposed algorithms is in the way in which the subgroups are discovered. Our proposed algorithms find a correlated user-item

subgroup for each user which is used in the prediction by using the least squares technique while MCoC and SCoC algorithms find several correlated subgroups at once and the prediction is performed by employing some CF algorithm in each subgroup. The final prediction is obtained by combining the results from all subgroups. We employ one of the most popular collaborative filtering algorithm known as the Maximum Margin Matrix Factorization (MMMF) with the subgroups discovered by MCoC to find list of top N recommendation items.

- *MCoC model with inner relationship (MCoC-UU-II-UI)*:
  Jiajun Bu et. al. [12] extended the subgroup based MCoC algorithm [68] by analyzing the inner relationships of the user-item interaction. The author analyzed the interactions simultaneously from three different directions such as user-user(UU), item to item (II) and user to item (UI). Once the similar user-item subgroups are identified, *MMMF* is used as the base collaborative filtering model to predict the missing ratings.

4.3.4 Results and analysis

To show the robustness of our subgroup based proposed algorithm, we compare our results with widely known state-of-the-art algorithms in terms of accuracy of the prediction of missing ratings and prediction of recommendation. The comparisons based on the two different types of accuracy are discussed separately.

4.3.4.1 Analysis of Prediction accuracy

We measure the prediction accuracy of our proposed algorithm by using well known state-of-the-art prediction algorithms such as item-based CF model [55] and a population based traditional CF model such as PSO-MMMF [19]. We use the popular accuracy metrics such as RMSE and MAE to evaluate the prediction quality of the four proposed algorithms *SLLSimCF*, *GA-SLLSimCF*, *ISLLSimCF*, *I-GA-SLLSimCF*. The most suitable algorithm among the proposed algorithms is selected to further compare with the existing traditional CF algorithms in terms of measuring the prediction accuracy. Firstly, we present the comparisons of all the proposed algorithms in terms of RMSE and MAE in tables (4.1,4.2) and highlight the most optimal result among

the proposed algorithms. In tables (4.1,4.2), our findings suggest that the algorithms that consider the user-item subgroups in prediction of missing ratings have a significant improvement on accuracy measures with respect to the base algorithm *LLSimCF*. The reason for significant improvement of the proposed algorithms is due to the consideration of only highly correlated user-item subgroups in computing prediction. The base algorithm, on the other hand, considers rating information of the entire items of similar users in the prediction of missing ratings. Therefore, our proposed algorithm justifies the effectiveness of consideration of only correlated user-item subgroup which excludes irrelevant items in the prediction computation. The performance of the base algorithm clearly shows that inclusion of rating information of the irrelevant items of similar users in prediction deteriorates the accuracy of the CF model. Among the several subgroups based proposed algorithms, the GA-based algorithm *GA-SLLSimCF* outperforms all the other proposed algorithms.

The superiority of *GA-SLLSimCF* among all the proposed algorithms is due to the effectiveness of the fitness function that is able to find the most correlated user-item subgroups from a large number of feasible solutions in the search space. The fitness function is an evaluation score of a user-item subgroup by using mean squared residue. The graph shown in figure (4.3e) shows the values of the fitness function throughout the evolutionary computation in GA based algorithm *GA-SLLSimCF* for an active user. The fitness function of *GA-SLLSimCF* is a minimization problem, therefore the graph drops gradually over the generations justifying the improvement of the discovery of better user-item subgroup successively. On the other hand, the two proposed iterative based subgroup algorithms which are *ISLLSimCF* and GA oriented *I-GA-SLLSimCF* showed more promising results than the base algorithm because of the inclusion of highly relevant user-item information in the prediction step. Although the iterative based GA oriented subgroup algorithm *I-GA-SLLSimCF* applies enhanced framework in the selection of different k number of similar users for each active user, it fails to outperform *GA-SLLSimCF* because of higher computational consumption and premature convergence. Note that both the iterative based proposed algorithms are highly more expensive in computational time than the non-iterative proposed algorithms *SLLSimCF*, *GA-SLLSimCF*. Also note that the effectiveness of the iterative based proposed subgroup algorithms with respect to the base algorithm can still be seen from the observations shown in tables (4.1,4.2). The additional execution time of *I-GA-SLLSimCF*

Size	LLSimCF	SLLSimCF	GA-SLLSimCF	ISLLSimCF	I-GA-SLLSimCF
10k	1.2804	1.2239	**1.1498**	1.2237	1.1511
50k	1.2547	1.1900	1.1487	1.1879	**1.1451**
100k	1.2297	1.1235	**1.0745**	1.1231	1.1182

Table 4.1: *RMSE* of the proposed algorithms for the benchmark dataset, *Movielens*.

Size	LLSimCF	SLLSimCF	GA-SLLSimCF	ISLLSimCF	I-GA-SLLSimCF
10k	0.9875	0.9300	**0.8312**	0.9290	0.8580
50k	0.9469	0.8929	**0.8289**	0.8912	0.8388
100k	0.9150	0.8371	**0.7946**	0.8369	0.8270

Table 4.2: *MAE* of the proposed algorithms for the benchmark dataset, *Movielens*.

is consumed in three tasks. Firstly, in discovering different k similar users for each active user assuming that each user need not have the same number of similar users k. Secondly, in discovering a subset of similar items from all the consumed items of the active user through GA for each active user. Finally, the predicted matrix is given as input to the successive iterations to enhance the prediction of the user-item rating matrix. However, the effectiveness of the iterative procedure in the proposed algorithms can be seen from the tables (4.1,4.2) by comparing with the base model and the non-iterative based algorithm *SLLSimCF*. The main reason behind the failure of *I-GA-SLLSimCF* is the premature convergence of the algorithm.

Table (4.3) presents the experiments of the proposed algorithm *GA-SLLSimCF* with different similarity measures such as Pearson correlation (PC), adjusted cosine (AC) and Euclidean distance (ED). The prediction computation derived from the similar group of users through ED outperforms the remaining two measures. We set the similarity measure to ED in the remaining experiments. Table (4.4) shows the prediction accuracy of the proposed subgroup based algorithm against the existing CF algorithms. The CF algorithms as given in the table like IB and PSO-MMMF takes into consideration the entire consumed items whereas in the case of the proposed subgroup based approach this is not the case. The impact of ruling out irrelevant information can be seen from the improvement in the values of RMSE and MAE.

Size	RMSE			MAE		
	PC	AC	ED	PC	AC	ED
10k	1.1969	1.1415	**1.1498**	0.8985	0.8710	**0.8312**
50k	1.1490	1.1578	**1.1487**	0.8434	0.8480	**0.8289**
100k	1.2228	1.1209	**1.0745**	0.8946	0.8182	**0.7946**

Table 4.3: Results related to different similarity measures of the proposed algorithm on benchmark dataset, *Movielens*.

Size	RMSE			MAE		
	Proposed	Base LS	Item-Based[55]	Proposed	Base LS	Item-Based[55]
10k	**1.1498**	1.2804	1.211	**0.8312**	0.9875	0.8440
50k	**1.1487**	1.2547	1.185	**0.8289**	0.9469	0.8304
100k	**1.0745**	1.2297	1.0759	**0.7946**	0.9150	0.7978
1M	**1.071**	1.1092	1.073	**0.7903**	0.8712	0.7929

Table 4.4: Comparison of the proposed algorithm with base LLSimCF and item-based CF [55] on benchmark dataset, *Movielens*.

CF models	*10k dataset*			*50k dataset*			*100k dataset*		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
MMMF	0.186	0.0439	0.0710	0.152	0.039	0.062	0.1580	0.0355	0.0579
SCoC-MMMF	0.162	0.0404	0.0646	0.156	0.0337	0.0554	0.154	0.0337	0.0553
MCoC-MMMF	0.198	0.0436	0.0715	0.184	0.0389	0.064	0.1700	0.0499	0.0771
MCoC-UU-II-UI	0.192	0.0259	0.0456	0.192	0.0206	0.0372	0.182	0.0252	0.0443
LLSimCF	**0.202**	0.0434	0.071	0.189	0.0425	0.0693	0.1750	0.0478	0.075
GA-SLLSimCF	0.174	0.0408	0.0661	**0.196**	**0.048**	**0.077**	**0.1999**	**0.0656**	**0.099**
IB	0.198	**0.0449**	**0.0732**	0.180	0.0427	0.069	0.156	0.0355	0.0578

Table 4.5: Comparison of the proposed algorithm in terms of recommendation accuracy

4.3.4.2 Analysis of Recommendation accuracy

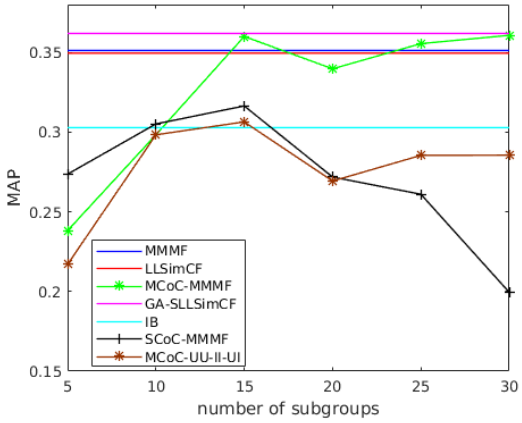
To measure the recommendation accuracy of the proposed algorithms, we compared it with several well known algorithms. Popular co-clustering algorithms such as multi-class (*MCoC*), single-class (*SCoC*) [68] and extended multiclass co-clustering algorithm with three inner relationships (*MCoC-UU-II-UI*) [12] were used for the comparison. We also made use of traditional algorithms such as item-based CF model [55] and *MMMF* [52] to evaluate top N recommendation accuracy.

We use popular recommendation accuracy metrics such as precision, recall, F1-score and MAP. Table (4.5) shows the comparisons of our proposed algorithm *GA-SLLSimCF* with the state-of-the-art algorithms in terms of recommendation accuracy metrics such as precision, recall and F1-score by using different dataset sizes. Our findings in table (4.5) suggest that the subgroup based CF models such as our proposed algorithm *GA-SLLSimCF* and subgroup based state-of-the-art algorithm *MCoC-UU-II-UI* outperform the corresponding base models which are *LLSimCF*, *MMMF* and IB though there is an exception with smaller datasets. The importance of consideration of only highly correlated user-item subgroup in prediction of missing ratings is justified for larger datasets. Although the single-class co-clustering algorithm, *SCoC-MMMF*, is based on subgroup, the performance of *SCoC-MMMF* deteriorates from the base MMMF as it does not allow the same user/item to belong to multiple subgroups by limiting their dispersive preferences. Based on our observations shown in the table (4.5), our proposed algorithm *GA-SLLSimCF* outperforms the well known algorithms, namely item based (IB) [55], co-clustering based multi-class algorithm *MCoC-MMMF* [68], the extended *MCoC-UU-II-UI*[12] and the base MMMF [52] in terms of bigger datasets such as 50k and 100k. Note that the observation gets clearer in the larger dataset. Our experiments suggest that the proposed algorithm fails to find highly correlated subgroups for smaller datasets since the number of consumed items to construct a subset of items is not sufficient to train the model accurately. In order to summarize the accuracy metrics Precision and Recall into single-figure metric for comparison, we use Mean Average Precision (MAP) metric that measures the quality across completeness of the recommended items which is shown in figures (4.4a,4.4b,4.4c). The model that is having a larger MAP value represents the better model.

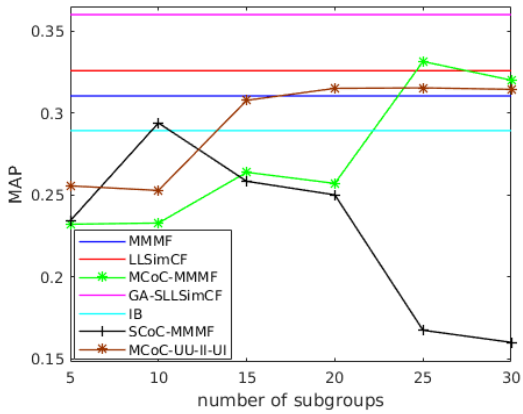
We observed that our proposed algorithm outperforms all the CF models under comparison in all the three datasets. Our findings in terms of MAP suggest that the correlation of the subgroups discovered by GA that evaluates fitness by calculating mean squared residue and row variance is significant though the effectiveness is less for smaller dataset like 10k. The low performance of multiclass co-clustering algorithms, *MCoC-MMMF*[68] and *MCoC-UU-II-UI*[12], in smaller number of subgroups shown in figures (4.4a,4.4b,4.4c) suggests that a subgroup consisting of a large number of users and items have more irrelevant information than the subgroups with lesser number of users and items which are empowered by allowing more overlapping due to the matching preferences of the users. Whereas the declining performance of single-class co-clustering algorithm, *SCoC-MMMF*, with the increasing number of subgroups suggests that one's preference is focusing only in few topics/subgroups rather than dispersive in different subgroups. Since the performance of the co-clustering algorithm *MCoC-MMMF* is unstable with different number of subgroups, we fix the number of subgroups to a value that trades off between the two constraints which turns out to be 15. Note that our base algorithm *LLSimCF* outperforms all the traditional CF such as item-based [55] and *MMMF*[52] in the figures (4.4b,4.4c) showing the accurate predictive behaviour of the LS model. The proposed algorithm outperforms both the popular subgroup based CF models, MCoC and extended MCoC-UU-II-UI, signifying that the user-item subgroups retrieved by the proposed algorithm which is a population based optimization algorithm, namely Genetic Algorithm (GA), is highly correlated comparing to the subgroups retrieved by fuzzy c-means of MCoC and MCoC-UU-II-UI models. Hence, more accurate preferences of the users gathered in the subgroup leads to more accurate prediction than the MCoC and MCoC-UU-II-UI. The superiority of the proposed algorithms comes from the capability of capturing highly correlated user-item subgroups by GA which ultimately improves the prediction.

4.3.4.3 Performance

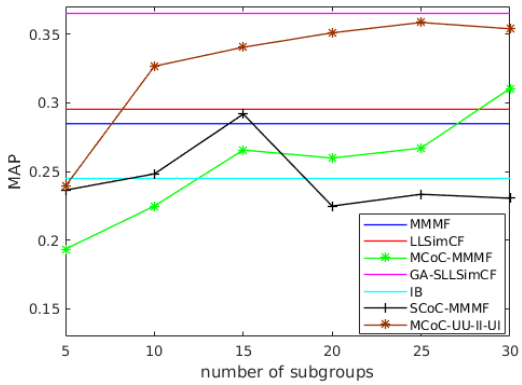
The computational complexity of finding k similar users is $O(n^2)$. So, the proposed algorithm takes approximately $O(n^3m)$, neglecting the complexity of the genetic algorithm as it is $O(max - iteration.popS)$ where $popS$ denotes the population size. Both the $max - iteration$ and $popS$ are finite and lesser than $O(nm)$ which is the cost of



(a) Comparisons of 10k dataset



(b) Comparisons of 50k dataset



(c) Comparison of 100k dataset

Figure 4.4: Recommendation performance comparison of different CF models based on *Mean Average Precision* (MAP) with respect to different number of subgroups.

prediction of maximum m missing items for n users. The computational complexity of the proposed algorithm outperforms *MCoC* as the *MCoC* takes $O(n^4m)$ wherein it takes $O(n^2)$ for mapping the matrix to low-dimensional space, $O(n+m)$ to find user-item subgroups and $O(nm)$ to find the recommendation score of the matrix. On the otherhand, the compuational complexity of the traditional Item-Based *IB* model takes $O(nm^3)$. The comparison of the proposed algorithm and the IB model depends on the size of each dataset.

4.4 Summary

Traditional collaborative filtering algorithms for recommender systems rely on the rating information of all the consumed items of similar users for predicting the missing ratings. One drawback of this traditional approach is that there is a possibility of including irrelevant items in the prediction process and this may lead to poor performance of the algorithms. To address this problem, we proposed a set of collaborative filtering algorithms that makes use of the concept of subgroups wherein a subset of correlated items based on a set of similar users is taken into consideration in computing the prediction. The prediction of missing ratings is performed by using the least squares approach. We make use of the evolutionary algorithm to search for the highly correlated user-item subgroup from a given large user-item rating matrix. The remarkable capability of genetic algorithm in balancing between exploration and exploitation of large search space gives a significant improvement in discovering highly correlated user-item subgroups. The users and items may belong to multiple subgroups implying that the preferences of the users are not focusing on few features of a subgroup, yet not as diverse as it would cover the entire available features which is not realistic. The rating information of the discovered user-item subgroup is used for effective prediction of the missing ratings. Our analysis pf the proposed methods suggest that *GA-SLLSimCF* is the most suitable algorithm among all the proposed algorithms. In order to perform a fair evaluation of the proposed algorithm (GA-SLLSimCF), we compare it with some of the state-of-the-art algorithms namely maximum margin matrix factorisation, item based CF as well as popular subgroup based algorithms such as multi-class co-clustering model (*MCoC*) and extended multi-class co-clustering (*MCoC-UU-II-UI*). Our extensive experiments using the proposed method demonstrates that the efficiency

in the prediction of missing ratings can be improved by considering the user-item subgroups rather than ratings of the entire set of items. For our experiments we use benchmark datasets like the *Movielens dataset* and various accuracy metrics like RMSE, MAE, precision, recall, F1-score and MAP are employed to demonstrate the robustness of the proposed algorithm (*GA-SLLSimCF*) in terms of prediction and recommendation accuracy. Our results highlight the superiority of including only the correlated user-item subgroups while making prediction in collaborative filtering models and this can help in further enhancing the CF models.We conclude from our experiments that subgroup based CF models are more accurate and meaningful when compared to algorithms that consider the entire set of consumed items.

Chapter 5

Discovery of User-Item Subgroups through Co-Clustering and Metaheuristics in Neighbourhood based Collaborative Filtering

In the previous chapter, we presented a framework for discovering user-item subgroups in collaborative filtering through a two-step process wherein the first process finds a set of similar users based on the entire items by using a similarity measure and then the second process discovers a highly correlated subset of items based on the selected subset of users. The corresponding algorithms and experimental setup with respect to the two-step process was also proposed in the previous chapter. In this chapter, we propose a method that identifies the correlated user-item subgroups in one step by employing an advanced clustering technique, called co-clustering. The co-clustering technique [27] is an extension of the traditional clustering technique which performs clustering of the data points simultaneously on both the dimensions of a matrix to capture local similarity structure, instead of grouping the data-points separately in each dimension as in the case of typical clustering. The co-clustering technique generates a subgroup of rows and columns of the matrix which exhibits local similar behaviour. We employ *Fuzzy C-Means (FCM)* clustering to perform the co-clustering, and the result of FCM provides highly correlated subgroups. To optimize the search of FCM, we employ *Particle Swarm Optimization (PSO)* that enhances the performance of FCM

by exploiting the globalised searching behaviour of PSO. Our objective is to discover optimal centroids through PSO and initialize the centroids obtained by PSO as the initial centroids in FCM. Therefore, FCM discovers meaningful user-item subgroups wherein the rating information of the subgroup is used in recommendation computation. Bin Xu et. al.[68] and Jiajun Bu et. al. [12] formulated Multiclass Co-Clustering (MCoC) problem and proposed a solution that employs meaningful user-item subgroup in prediction of recommendation items. We adopted the MCoC framwork to discover the highly correlated user-item subgroups and integrated PSO to optimize the search process for finding the optimal solution.

5.1 Co-Clustering

Co-clustering technique was first introduced by Hartigan [27] in 1972 as a direct clustering technique wherein simultaneous clustering on both rows and columns are done. It is a variation of the traditional clustering that aims to identify local similarity structure of a data matrix. Co-clustering technique is well studied in microarray data analysis and text mining for handling sparsity and dimensionality reduction problems. Several researchers in the area of collaborative filtering also explored co-clustering for solving issues in recommender systems from different aspects [5, 9, 17, 32, 36, 42, 61, 70].

The objective of Co-Clustering in collaborative filtering is to simultaneously group both the users and items into c similar co-clusters/subgroups from $Y \in \mathbb{R}^{n \times m}$ user-item rating matrix in which there are n users and m items. Each subgroup is formed based on the similarity degree of the users and items to each other. The Multiclass Co-Clustering (MCoC) technique [68] is adopted as the base co-clustering technique in our proposed algorithm. The proposed algorithm employs Fuzzy C-Means (FCM) to find subgroups of similar users and items. In FCM, a partition matrix, $P \in [0,1]^{(n+m)\times c}$, is used to represent membership degree of the users in the respective subgroups. The partition matrix P is represented as given in (5.1):

$$P = \begin{bmatrix} Q \\ R \end{bmatrix} \tag{5.1}$$

where $Q \in [0,1]^{n\times c}$ and $R \in [0,1]^{m\times c}$ represent the partition matrices of users and items respectively. The resulted partition matrix P represents the solution of the base co-clustering technique, which means that the subgroups formed are the optimal solutions to further make an accurate prediction. The property of the partition matrix is given in Eqn. (5.2).

$$P_{i,j} = \begin{cases} P_{i,j} > 0, & \text{if the } i^{th} \text{ entry belongs to the } j^{th} \text{ subgroup} \\ P_{i,j} = 0, & \text{Otherwise} \end{cases} \tag{5.2}$$

Each element $P_{i,j}$ denotes membership degree of an entry i of the partition matrix P to the j^{th} subgroup. The membership degree of an entry i of the matrix P for each subgroup sums to one. Each entry belongs to a fixed number of subgroups such that $1 \leq k \leq c$ wherein each entry is allowed to belong to k subgroups. Note that when $k = 1$, the technique works as a traditional clustering algorithm. The aim of the base co-clustering is to map all the users and items to low dimensional space while preserving the original information, and search correlated user-item subgroups to discover relevant items for recommendation. The base co-clustering technique consists of two main stages which are explained below.

Step 1: *Application of Fuzzy C-Means clustering.*

Step 1.1: *Map all the users and items to low dimenensional space.*

The optimal z-dimensional solution X that preserves preference information can be obtained from the optimization function[68] which is given in (5.3):

$$\min_{X} \quad Tr(X^T N X) \tag{5.3}$$

such that

$$X \in \mathbb{R}^{(n+m)\times z}, X^T X = I, \qquad N = \begin{bmatrix} I_n & -S \\ -S^T & I_m \end{bmatrix}, \qquad S = (G^{row})^{-1/2} Y (G^{col})^{-1/2}, \tag{5.4}$$

where I_n and I_m correspond to the identity matrices of size $n \times n$ and $m \times m$ whereas $G^{row} \in \mathbb{R}^{n\times n}$ and $G^{col} \in \mathbb{R}^{m\times m}$ correspond to diagonal degree matrices of users and items with $G^{row}_{i,i} = \sum_{j=1}^{m} Y_{i,j}$ and $G^{col}_{j,j} = \sum_{i=1}^{n} Y_{i,j}$. The optimization function given in (Eqn. 5.3) is minimized by the optimal solution X which is an eigenvalue solution of all the users and items such that $NX = \lambda X$ where $X = [x_1, x_2, \cdots, x_a]$ such that

$x_1, \cdots, x_a$ are the least eigenvectors of matrix N for the respective sorted eigenvalues.

Step 1.2: *Find correlated user-item subgroups.*

The optimal partition matrix P can be obtained by finding correlated subgroups from the unified user-item data X. The user-item subgroups can be discovered in two ways: either single class or multi-class based on the factor of overlapping among the subgroups. The k-means clustering is used to discover the subgroups from the data X in single class co-clustering (*SCoC*). Each entry of P in *SCoC* belongs to only one subgroup whereas fuzzy c-means (FCM) clustering is used to find subgroups for multi-class co-clustering. FCM works by assigning fuzzy membership value to each data object corresponding to each cluster and updates the membership value until the algorithm converges to an optimal solution. The optimization problem of FCM is a minimization function [68] which is given as in (5.5).

$$J_m(P,V) = \sum_{i=1}^{m+n}\sum_{j=1}^{c} P_{ij}^{l} d(x_i, v_j)^2 \tag{5.5}$$

where P_{ij} represents membership value of the entry x_i to the j^{th} cluster, v_j represents cluster centroid while V represents matrix of the cluster centroids. l represents the weighting exponent that controls the fuzziness of the partition and d is the distance function wherein in this paper we use the Euclidean distance. The partition matrix and cluster centroids are updated in every iteration by making use of the equation as given in (5.6).

$$P_{ij} = \frac{d(x_i, v_j)^{\frac{2}{1-l}}}{\sum_{k=1}^{c}(d(x_i, v_k))^{\frac{2}{1-l}}}, v_j = \frac{\sum_{i=1}^{m+n} P_{ij}^{l} x_i}{\sum_{i=1}^{m+n} P_{ij}^{l}}, \qquad \forall i = 1, \cdots, (n+m); \forall j = 1, \cdots, c. \tag{5.6}$$

The algorithm is terminated when an improvement in the value of the optimization function at two consecutive iterations is smaller than the threshold ϵ.

Step 2: *Recommendation.*

Once correlated user-item subgroups are discovered, any existing collaborative filtering algorithm can be applied in each subgroup to predict the missing ratings. The results from each subgroup are merged to get a unified prediction score for recommendation. The unified method [68] that maintains all the cases including the overlapping of the subgroups is given in Eqn. (5.7). Let $Pred(u_i, y_j, k)$ denote the prediction score

obtained by a user i on item j in a subgroup k, and Z_{ij} denote the unified prediction score of user i on item j. Then the prediction score Z_{ij} can be given as

$$Z_{i,j} = \begin{cases} \sum_k Pred(u_i, y_j, k).\delta_{ik} & \text{if } u_i \text{ and } y_j \text{ belong to one or more subgroups,} \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

where δ_{ik} represents an indicator value of user i that corresponds to the most interesting subgroup k among the overlapped subgroups with item j. Now, δ_{ik} can be described as in Eqn. (5.8).

$$\delta_{ik} = \begin{cases} 1 & \text{if } P_{ik} \text{ is max}(Q_{u_i} \cap R_{y_j}), \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

5.2 Proposed Algorithm

In the previous section, we described the base co-clustering technique that is applied in collaborative filtering to identify highly correlated user-item subgroups in the rating matrix by using Fuzzy C-Means (FCM) clustering. FCM algorithm starts with a partition matrix and c clusters in which the initial centroid of the clusters are randomly initialized and then the cluster centroids and partition matrix are subsequently updated in further iterations. However, the performance of FCM algorithm has high impact on the initial position of the centroids being selected and may get trapped in local optima if it is not rightly chosen. Thus, we employ a metaheuristic optimization algorithm, namely Particle Swarm Optimization (PSO), to discover the nearest optimal centroid for each cluster to be initialized as the initial centroids in the FCM component rather than initializing randomly as in the case of previous section. In PSO, the searching technique starts with a swarm of candidate solutions in the high dimensional search space, such that the fitness of each candidate solution is computed and the best candidate solution is utilized to influence the remaining solutions towards the optimal solution.

In this chapter, we propose a novel hybrid framework of Particle Swarm Optimization and Co-Clustering, named as *PSO-CoC*. The goal of the proposed algorithm is to find highly correlated user-item subgroups from the whole user-item rating matrix. This is done by initializing the clusters' centroids of FCM to the nearest optimal centroids found by PSO in collaborative filtering. The block diagram of the proposed

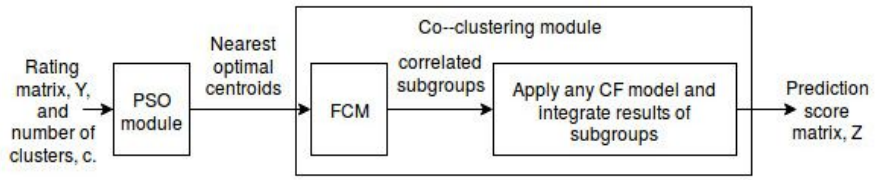


Figure 5.1: Block diagram of the proposed method, PSO-CoC

framework is shown in Fig.(5.1). As shown in Fig.(5.1), PSO module is fed with known user-item rating matrix, Y, and number of clusters, c, as inputs. The PSO module discovers nearest optimal centroids of the clusters effectively by utilizing collective intelligence exhibited by the multiple particles, and feeds the nearest optimal centroids as initialized inputs to the co-clustering module. FCM technique in the co-clustering module starts grouping similar subsets of users and items by initializing the initial centroids of FCM to the nearest optimal centroids discovered by the PSO module. Over the iterations, the position of centroids and partition matrix are updated to a better solution by minimizing the objective function given by Eqn.(5.5). Finally, FCM converges to highly correlated subgroups of users and items, and FCM allows overlapping of the subgroups as well. After FCM converges, a traditional CF algorithm is executed in each subgroup to predict the missing ratings of the items in the subgroup by utilizing the rating information of only the subgroup. To unify the ratings predicted for an item in different subgroups, a unified prediction score is evaluated by using Eqn.(5.7).

PSO module: In our proposed algorithm, each particle is represented by a matrix $\{v_1, v_2, \cdots, v_c\}$, where c is the number of clusters and v_i denotes the centroid vector of the i^{th} cluster wherein a is the dimension of the optimal solution X. The representation of a particle is given below.

$$particle = \begin{pmatrix} v_1 \\ v_2 \\ \cdots \\ v_c \end{pmatrix} = \begin{pmatrix} v_{11} & v_{12} & \cdots & \cdots & v_{1a} \\ v_{21} & v_{22} & \cdots & \cdots & v_{2a} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ v_{c1} & v_{c2} & \cdots & \cdots & v_{ca} \end{pmatrix}$$

The fitness degree of each particle is evaluated by finding the average distance between a cluster and a user, which is given by Eqn. (5.9). Thus, smaller value of the fitness

function signifies stronger correlation of the cluster [16].

$$f = \frac{\sum_{i=1}^{c} \frac{\sum_{j=1}^{q_i} d(v_i, u_{ij})}{q_i}}{c} \tag{5.9}$$

where q_i denotes the number of users which belongs to cluster i; u_{ij} represents the j^{th} user vector that belongs to cluster i; v_i is the centroid vector of i^{th} cluster; $d(v_i, u_{ij})$ represents distance between centroid vector of the cluster i and user vector j that belongs to the cluster i.

A particle is represented as $particle_i(t)$ where i denotes particle i at time step t, and velocity vector i at time step is represented as $vel_i(t)$. The particle and velocity vectors are updated by using the Eqn. (2.14 and 2.15) respectively as given in Chapter 2. The proposed algorithm is described in Algorithm (9).

5.3 Experimental Settings and Results

5.3.1 Experimental Setup

We conducted extensive experiments of our proposed algorithm on a benchmark dataset, 100k *Movielens* in which there are 100,000 ratings given by 945 users on 1682 items. The dataset is a sparse matrix having 93.71% sparsity. The missing ratings are filled with zero and the other ratings of the dataset range from 1 to 5 such that 1 indicates the lowest rating value and 5 denotes the highest. The best model is achieved by tuning the parameters of the problem to suitable values. Quality of an algorithm has a lot of impact on the value of the parameters being chosen. The selected parameter values corresponding to each module of the proposed algorithm are described below.

Parameters of PSO module

- The swarm size of PSO module is set to 50.
- The uniformly distributed random values, $rand1$ and $rand2$, are set to 0.1270 and 0.0975 respectively.
- The acceleration coefficients c_1, c_2 and inertia weight w are set to 1.42, 1.42 and 0.72 respectively.

Algorithm 9: PSO-CoC

Input: User-item rating matrix Y, and number of clusters c.
Output: Prediction score matrix Z
//Start PSO Module//
Initialize swarm with randomly selected k user vectors from matrix Y as cluster centroids;
while *(max-iteration is not met)* **do**
    for *each particle* **do**
        Assign each user vector to the nearest cluster centroid;
        Evaluate fitness function by using (5.9);
        Update position and velocity of the particle based on (2.14, 2.15) to generate more optimal solution;
//Start Co-clustering Module//
Initialize initial centroids of FCM with the nearest optimal centroids found by PSO module;
while *(termination condition is not satisfied)* **do**
    Compute objective function,J, which is given by (5.5);
    Update partition matrix, P, and centroid of the clusters, V, given by (5.6).
Apply any CF algorithm to each subgroup, and predict missing values in each subgroup;
Apply the unified framework given by (5.7) to compute the prediction score matrix, Z ;
Find top N item recommendations and compute accuracy of the algorithm;

- The user-defined termination condition of PSO module is set to 50.

Parameters of Co-clustering module

- Weighting exponent of FCM, l is set to 2.
- Threshold ε to 0.00001.
- The user-defined termination condition of FCM is set to 50.
- Number of subgroups, k, that a user is allowed to overlap is $k = log_2(c)$ where c is the total number of subgroups [68].

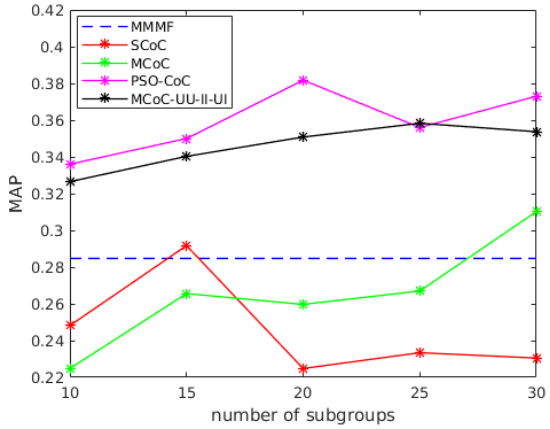
The proposed algorithm is evaluated by using popular accuracy metrics for top N recommendation such as *precision, recall, F1-score* and *Mean Average Precision (MAP)*. N is set to 10 in all the experiments.

Comparison

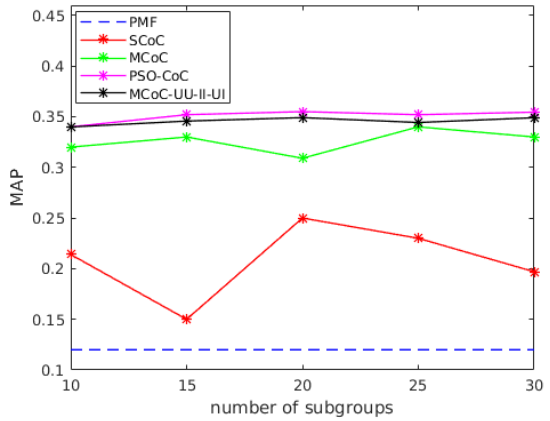
To measure the effectiveness of our proposed algorithm, we selected popular user-item subgroup based algorithms like Multiclass Co-Clustering (*MCoC*) algorithm, Single-class Co-Clustering algorithm (*SCoC*)[68] and an extended version of the Multiclass Co-Clustering (*MCoC*) algorithm which gives more insight about the user-item relationship [12]. The extended algorithm is named as *MCoC-UU-II-UI*. The above mentioned algorithms aims at utilising user-item subgroups in the prediction of a rating matrix for top N recommendation. We employ a few state-of-the-art collaborative filtering algorithms such as Maximum Margin Matrix Factorization (*MMMF*)[52], Probabilistic Matrix Factorization (*PMF*)[53] and Item-Based (*IB*)[55] algorithms as the base collaborative filtering models in our proposed algorithm.

5.3.2 Experimental Results and Discussions

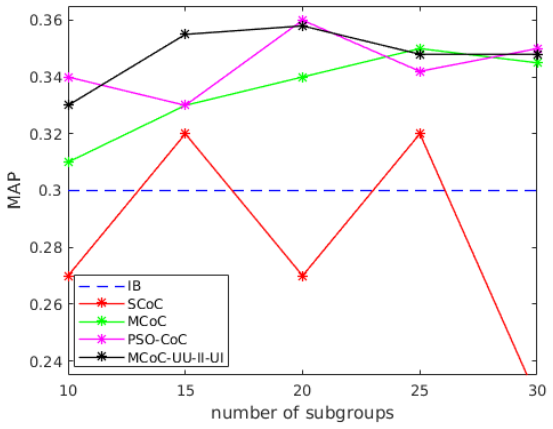
The comparison of the proposed algorithm *PSO-CoC* with different recommendation algorithms such as *MMMF*, *PMF* and *IB* which are used as the base collaborative filtering models and the application of the different base CF models with popular subgroup



(a) MMMF



(b) PMF



(c) IB

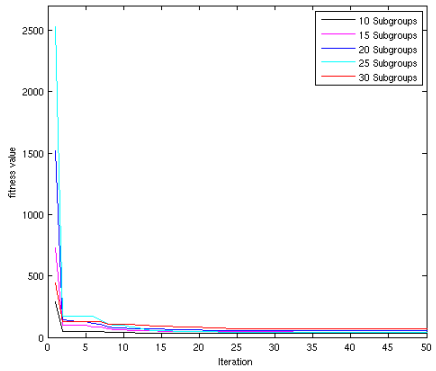
Figure 5.2: Performance of the proposed algorithm (*PSO-CoC*) with three well known CF models across different number of subgroups.

based models such as *MCoC*, *SCoC*, *MCoC-UU-II-UI* [12, 68] for top N recommendations are shown in Tables (5.1,5.2,5.3) and Fig. (5.2). In Fig. (5.2), the comparisons are made in terms of *Mean Average Precision* (MAP) across different *number of subgroups* for all the algorithms. The performance of the base CF models are represented by dark blue bar in the figure. We observe from the figure that single class co-clustering model performs better when the number of subgroups are lesser and the performance level falls with an increase in the number of subgroups. As the number of subgroups increases, the number of items contained in a subgroup decreases. This shows overspecification of the features in the subgroups. Thus, accuracy drops in the case of single class model with higher number of subgroups due to the restriction of *overlapping* among the subgroups that consequently disable identifying meaningful subgroups. In the case of multi-class subgroup based models such as our proposed algorithm (*PSO-CoC*), *MCoC* and *MCoC-UU-II-UI* performance is better with higher number of subgroups as compared to lower number of subgroups and this solves the *preference over specification* problem. In addition to that, allowing overlapping of subgroups helps the models to predict accurately by using the rating information of the preferences from highly relevant multiple subgroups. The advantage of overlapping among the subgroups is to boost the possibility of capturing user's preferences by removing irrelevant information. The reason why multi-class models do not perform well with lesser number of subgroups is because of the incapability of extraction of highly similar interests from the large number of choices available in the subgroup. From Tables (5.1,5.2,5.3) and Fig. (5.2), we observe that the multi-class models outperform both the single class models and the base algorithms in all the cases of different base CF methods. Our findings suggest that the correlated user-item subgroups influence the accuracy of the recommender system in prediction computation. The superiority of the proposed algorithm from the existing well known subgroup based CF model is the intialization of the centroids of FCM to the nearest optimal centroids generated by the PSO instead of initializing randomly at any point of the hyper-dimensional search space. The initialization of the centroids to optimal positions helps the proposed framework to do better prediction.

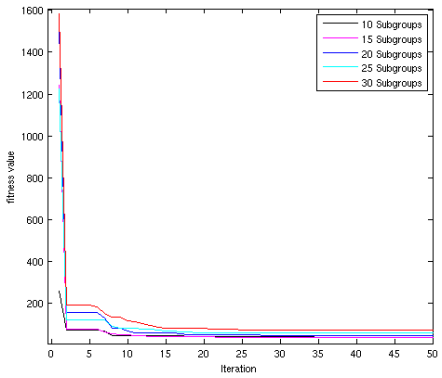
Apart from *Mean Average Precision*, the proposed algorithm (*PSO-CoC*) is further measured for accuracy by using popular metrics for top N recommendation such as *Precision*, *Recall* and *F1-score*. The proposed algorithm is compared for two values

of k, the number of subgroups, and the results are shown in Tables (5.1,5.2,5.3). It is evident from the results that the models with higher number of subgroups are able to capture more accurate fuzzy weights in clustering than the lesser number of subgroups, and the recommender system benefits from the participation of higher number of correlated subgroups in the prediction. The Tables (5.1,5.2,5.3) show that the subgroup based algorithms outperform the corresponding base CF algorithm, and also the multi-class algorithms outperforms the single class algorithms. Further, our proposed algorithm (*PSO-CoC*) improves the popular subgroup based algorithm *MCoC* by directing FCM clustering from optimized initial centroids which are discovered by PSO. The proposed algorithm also outperforms the extended multiclass co-clustering model *MCoC-UU-II-UI* with the impact of proper initialization of the centroids for FCM clustering. The results of Tables (5.1,5.2,5.3) suggest that the subgroups retrieved are highly correlated and hence influence the recommender system.

Figures (5.3a, 5.3b) represent the fitness graphs of the proposed algorithm for different base collaborative filtering models, such as *MMMF* and *IB*, with different number of subgroups. The fitness funtion is a minimization function and thus the graphs have a decreasing slope across the subsequent iterations. We observe a common characteristic in both the graphs of the figures that there is a rapid drop of the graphs in the beginning of the iterations, then gradually converge to the optimal solution and become stable. This means that the fitness function which is an average distance between the centroid of the cluster and the user vector is able to get to the nearby subspace of the optimal centroids after a few iterations which subsequently lead to the nearest optimal centroids of the clusters. We observe different patterns of fitness graphs for different number of subgroups in the figures. The graph with 10 number of subgroups achieved the lowest fitness value in the Fig. (5.3a) suggesting that the proposed algorithm converged immaturely with lower number of subgroups since the subgroups have large number of items which make it hard to capture only highly similar preferences. Fig. (5.3b) suggests that the proposed algorithm is not able to capture optimum fuzzy weights for larger number of subgroups in the beginning of the iteration but subsequently converged to the optimal solution. Thus, trade-off between the number of subgroups and fitness value is a factor in achieving an accurate recommender system. The performance of *PMF* as a base algorithm behaves similarly with *MMMF* Fig (5.3a) due to the similar characteristics of the matrix factorization.



(a) MMMF



(b) IB

Figure 5.3: Performance of the proposed algorithm (*PSO-CoC*) with different number of subgroups for base CF models.

Base MMMF	*Precision*		*Recall*		*F1-score*	
	0.158		0.0355		0.058	
Subgroup based CF models	*10 subgroups*			*20 subgroups*		
	Precision	*Recall*	*F1-score*	*Precision*	*Recall*	*F1-score*
SCoC	0.152	0.0198	0.035	0.155	0.0347	0.0564
MCoC	0.168	0.022	0.039	0.171	0.0399	0.0635
MCoC-UU-II-UI	0.18	0.0242	0.0426	0.192	0.0276	0.0481
PSO-CoC	0.178	0.0244	0.043	**0.21**	**0.0662**	**0.052**

Table 5.1: Comparison of the algorithms *MCoC*, *MCoC-UU-II-UI* and *SCoC* and the proposed algorithm (*PSO-CoC*) with base algorithm for predicting missing values as *MMMF*

Base PMF	*Precision*		*Recall*		*F1-score*	
	0.05		0.02		0.020	
Subgroup based CF models	*10 subgroups*			*20 subgroups*		
	Precision	*Recall*	*F1-score*	*Precision*	*Recall*	*F1-score*
SCoC	0.104	0.02	0.03	0.15	0.021	0.032
MCoC	0.174	0.026	0.045	0.175	0.027	0.046
MCoC-UU-II-UI	0.18	0.0262	0.0457	0.198	0.0276	0.0482
PSO-CoC	0.182	0.024	0.043	**0.202**	**0.035**	**0.06**

Table 5.2: Comparison of the algorithms *MCoC*, *MCoC-UU-II-UI* and *SCoC* and the proposed algorithm (*PSO-CoC*) with base algorithm for predicting missing values as *PMF*

Base IB	*Precision*		*Recall*		*F1-score*	
	0.8		0.0355		0.058	
Subgroup based CF models	*10 subgroups*			*20 subgroups*		
	Precision	*Recall*	*F1-score*	*Precision*	*Recall*	*F1-score*
SCoC	0.106	0.018	0.0313	0.11	0.019	0.032
MCoC	0.178	0.024	0.041	0.186	0.029	0.042
MCoC-UU-II-UI	0.19	0.0241	0.0439	0.20	0.0315	0.0544
PSO-CoC	0.19	0.025	0.044	**0.204**	**0.037**	**0.06**

Table 5.3: Comparison of the algorithms *MCoC*, *MCoC-UU-II-UI* and *SCoC* and the proposed algorithm (*PSO-CoC*) with base algorithm for predicting missing values as *IB*

5.4 Summary

We presented a hybrid framework of particle swarm optimization and fuzzy c-means clustering to discover meaningful user-item subgroups for collaborative filtering based recommender systems. The significance of utilization of user-item subgroups in CF for computing prediction ratings is demonstrated by comparing with different base CF models such as maximum margin matrix factorization, probabilistic matrix factorization and item-based models. The particle swarm optimization technique is used to drive the search direction of the initial centroids for fuzzy c-means to the optimal subspace by exploiting the powerful globalized searching behaviour of PSO. Then, fuzzy c-means is used to find highly correlated user-item subgroups wherein any original collaborative filtering is adopted to find prediction ratings of each user in the subgroup. A final prediction rating of each user is evaluated by using information from all mutual subgroups that the user belongs to and used to find top N recommendation items. In our proposed algorithm, the metaheuristic optimization algorithm acts as a booster to improve the fuzzy c-means clustering in discovering highly correlated user-item subgroups by initializing the initial centroid of the clusters to the nearest optimal solutions. Our experimental results have shown a promising way of user-item subgroups in helping to capture highly similar user preferences on a subset of items.

Chapter 6

Conclusions and Future Work

In this thesis our focus is to explore how metaheuristic techniques like Particle Swarm Optimisation (PSO) and Genetic Algorithm (GA) can be combined with collaborative filtering methods to address accuracy issues in recommender systems. We demonstrated how this combination can be carried out in both model-based collaborative filtering as well as neighbourhood-based collaborative filtering. In model-based collaborative filtering, we initially proposed an algorithm called GDPSO-MMMF that combined the maximum margin matrix factorisation (MMMF) based collaborative filtering model with particle swarm optimisation to find the optimal factored matrices U and V. In the second proposed algorithm (PSOm-MMMF), we optimized the GDPSO-MMMF framework by adding an operation called mutation which improves the swarm diversity and thus results in a better framework for finding the most optimal solution, i.e. the low rank factored matrices U and V. In the third algorithm (HPSO-MMMF), we proposed a hierarchical PSO based MMMF that finds the solution by following a tree-like structure where a parent with their immediate children forms a neighbourhood with the parent being the best particle in each neighbourhood. Finally, we propose an algorithm that combines genetic algorithm (GA) with MMMF with an aim to exploit the survival of the fittest approach of GA as well as helps in differentiating the search for an optimal solution as is done in techniques like PSO. Our experimental results on benchmark datasets demonstrate that when the exploration capability of population based search algorithms is combined with gradient search direction of MMMF, the proposed models are able to achieve better accuracy as can be evidenced from the derived RMSE and MAE values .

As far as neighbourhood-based collaborative filtering is concerned, our objective is to discover highly correlated user-item subgroups from a given matrix and then predict the missing ratings by making use of the rating information from the subgroups. Traditional collaborative filtering models are incapable of extracting highly correlated partial matching of the preferences among the users and items, and therefore cannot utilise such information in prediction or recommendation. We identify highly correlated user-item subgroups by removing irrelevant subset of items from group of similar users by using a two-step process unlike in typical neighbourhood based method that considers the entire information of similar users. We have outlined four different algorithms to identify only a subset of users and items which are highly relevant to each other by using a two step process. In the first step, the algorithms need to discover the highly correlated user-item subgroups by using one of the following techniques: 1) using a threshold to check the correlated degree of the subgroups, or 2) leverage metaheuristic approach like genetic algorithm in searching for highly correlated user-item subgroups, or 3) flexible number of similar users for each target user based on their similarity degree, or 4) a hybrid of the above. The four proposed algorithms differ based on which one of the four techniques as defined above is being used. In the second step the rating information from the highly correlated user-item subgroups are utilized in prediction computation of missing ratings through the least squares method. We have shown statistical performance of the proposed user-item subgroup based algorithms with three different similarity measures such as Pearson correlation, adjusted cosine and Euclidean distance. We compared our subgroup based proposed algorithms with the existing subgroup based collaborative filtering algorithm, namely MCoC, as well as state-of-the-art algorithms such as item-based neighbourhood method, MMMF, and metaheuristic based collaborative filtering algorithm such as PSO-MMMF. Experimental results show that our proposed algorithm outperforms all the CF models under comparison in all the three datasets. Our findings in terms of MAP suggest that the correlation of the subgroups discovered by GA that evaluates fitness by calculating mean squared residue and row variance is significant though the effectiveness is less for smaller dataset like 10k.

We also employed a co-clustering technique to discover correlated user-item subgroups in one go rather than adopting a two step process. The co-clustering technique find close associations among users and items and form subgroups in each iteration.

To perform co-clustering technique, we adopted Fuzzy C-Means (FCM) clustering in the proposed algorithm. The objective function of FCM that determines closeness of user-item associations is average distance of intraclass relationships among the users in the clusters. The similarity in subgroups gets better over the iterations through the characteristic of learning rules in the technique for centroid of each cluster and partition matrix of FCM. We employ particle swarm optimization to find optimal positions for the initial centroid of co-clustering technique to start with. We have shown comparison of the proposed PSO based co-clustering algorithm with the underlined algorithm which uses only co-clustering to find the subgroups. In this framework, any traditional collaborative filtering algorithm can be employed to find missing ratings in the subgroups. Thus, we employed state-of-the-art algorithms like MMMF, PMF, item-based neighbourhood algorithms and combined with the proposed framework to judge the performances of the algorithms. We measured and compared the proposed algorithm in terms of top N recommendations.

In the future, we can include interclass relationship of users in the formulation of objective function in co-clustering technique. Note that we adopted intraclass relationship in the objective function of co-clustering in our proposed method to form correlated subgroups (Chapter 5). The quality of intraclass relationship of users in a subgroup can be evaluated by finding the distance between the users in the same subgroup which should be minimized. In addition to intraclass relationship, we could adopt interclass relationship wherein the distance between the users that belong to different subgroups should be far from each other, unlike intraclass relationships fr building subgroups of good quality. Note that the objectives of interclass and intraclass relationships conflict to each other to formulate in a single objective function as intraclass and interclass distances should be minimized and maximized respectively. Therefore, we can solve the problem by using Multi-Objective Clustering techniques. The Multi-Objective Clustering technique can be solved by using evolutionary approach and the technique can be considered as a multi-objective optimization problem in which we have a set of candidate clusterings and two different criterion functions which conflict to each other. We will get a set of pareto optimal solution that tradeoff the conflicting objective functions and the solutions in the pareto front are non-dominant to each other. The best compromised solution is chosen based on the decision makers. In addition, we can extend the work by employing different biclustering algorithms to discover

the highly correlated subgroups of users and items from the rating matrix. The more correlated the discovered subgroups are, the better the prediction of the collaborative filtering algorithms.

References

[1] **MovieLens**. `https://grouplens.org/datasets/movielens/`. (30)

[2] ASSAD ABBAS, LIMIN ZHANG, AND SAMEE U. KHAN. **A survey on context-aware recommender systems based on computational intelligence techniques**. *Computing*, **97**(7):667–690, 2015. (28)

[3] PANAGIOTIS ADAMOPOULOS AND ALEXANDER TUZHILIN. **On Unexpectedness in Recommender Systems: Or How to Better Expect the Unexpected**. *ACM Trans. Intell. Syst. Technol.*, **5**(4):54:1–54:32, December 2014. (38)

[4] SHAFIQ ALAM, GILLIAN DOBBIE, AND PATRICIA RIDDLE. **Towards Recommender System Using Particle Swarm Optimization Based Web Usage Clustering**. In LONGBING CAO, JOSHUA ZHEXUE HUANG, JAMES BAILEY, YUN SING KOH, AND JUN LUO, editors, *New Frontiers in Applied Data Mining*, pages 316–326, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. (2, 28)

[5] FARIS ALQADAH AND ET. AL. **Biclustering Neighborhood-based Collaborative Filtering Method for Top-n Recommender Systems**. *Knowl. Inf. Syst.*, **44**(2):475–491, August 2015. (2, 20, 90)

[6] SAGARIKA BAKSHI, ALOK KUMAR JAGADEV, SATCHIDANANDA DEHURI, AND GI-NAM WANG. **Enhancing scalability and accuracy of recommendation systems using unsupervised learning and particle swarm optimization**. *Applied Soft Computing*, **15**:21 – 29, 2014. (2, 21, 28)

[7] DANIEL BARLA-SZABÓ. *A study of gradient based particle swarm optimisers*. PhD thesis, University of Pretoria, Pretoria, 2010. (7, 23, 24, 25, 47)

[8] ALEJANDRO BELLOGIN AND JAVIER PARAPAR. **Using Graph Partitioning Techniques for Neighbour Selection in User-based Collaborative Filtering**. RecSys '12, pages 213–216, New York, NY, USA, 2012. ACM. (17)

[9] C. BIRTOLO AND ET. AL. **Improving accuracy of recommendation system by means of Item-based Fuzzy Clustering CF**. In *11th Intl Conf on Intelligent Systems Design and Applications*, pages 100–106, 2011. (90)

[10] PHILIP E. BOURNE, MICHAEL GRIBSKOV, RUSS B. ALTMAN, NANCY JENSEN, DEBRA A. HOPE, THOMAS LENGAUER, JULIE C. MITCHELL, ERIC D. SCHEEFF, CHRIS SMITH, SHAWN STRANDE, AND HELGE WEISSIG, editors. *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, August 19-23, 2000, La Jolla / San Diego, CA, USA*. AAAI, 2000. (69)

[11] JOHN S BREESE, DAVID HECKERMAN, AND CARL KADIE. **Empirical analysis of predictive algorithms for collaborative filtering**. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998. (2, 13, 17)

[12] JIAJUN BU, XIN SHEN, BIN XU, CHUN CHEN, XIAOFEI HE, AND DENG CAI. **Improving Collaborative Recommendation via User-Item Subgroups**. *IEEE Trans. on Knowl. and Data Eng.*, **28**(9):2363–2375, September 2016. (2, 20, 80, 84, 85, 90, 97, 99)

[13] ZHIPENG CAI, MAYSAM HEYDARI, AND GUOHUI LIN. **Iterated Local Least Squares Microarray Missing Value Imputation**. *J. Bioinformatics and Computational Biology*, **4**(5):935–958, 2006. (60, 61, 72)

[14] PABLO CASTELLS, NEIL J. HURLEY, AND SAUL VARGAS. **Novelty and Diversity in Recommender Systems**. In *Recommender Systems Handbook*, pages 881–918. 2015. (38)

[15] KIN-ON CHENG, NGAI-FONG LAW, AND WAN-CHI SIU. **Iterative bicluster-based least square framework for estimation of missing values in microarray gene expression data**. *Pattern Recognition*, **45**(4):1281–1289, 2012. (60, 61, 66)

[16] XIAOHUI CUI AND THOMAS E. POTOK. **Document Clustering Analysis Based on Hybrid PSO+K-means Algorithm**. *Special Issue*, pages 27–33, 2005. (95)

[17] P. A. D. D. C. DE CASTRO AND ET. AL. **Applying Biclustering to Perform CF**. In *Seventh International Conf on Intelligent Systems Design and Applications*, pages 421–426, 2007. (2, 3, 19, 20, 28, 90)

[18] MUKUND DESHPANDE AND GEORGE KARYPIS. **Item-based top-N Recommendation Algorithms**. *ACM Trans. Inf. Syst.*, **22**(1):143–177, January 2004. (17)

[19] V SOWMINI DEVI, KAGITA V RAO, ARUN K. PUJARI, AND VINEET PADMANABHAN. **Collaborative filtering by PSO-based MMMF**. In *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*, pages 569–574. IEEE, 2014. (xiv, 21, 28, 55, 57, 79, 80)

[20] ERNESTO DIAZ-AVILES, MIHAI GEORGESCU, AND WOLFGANG NEJDL. **Swarming to Rank for Recommender Systems**. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, pages 229–232, New York, NY, USA, 2012. ACM. (2, 28)

[21] M. DORIGO AND G. DI CARO. **Ant colony optimization: a new meta-heuristic**. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, **2**, pages 1470–1477 Vol. 2, July 1999. (22)

[22] ALAM ET AL. **Hierarchical PSO clustering based recommender system**. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8. IEEE, 2012. (2, 21, 28)

[23] GHOSH ET AL. **Hierarchical dynamic neighborhood based PSO for global optimization**. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 757–764. IEEE, 2011. (7, 49)

[24] FLORENT GARCIN, BOI FALTINGS, OLIVIER DONATSCH, AYAR ALAZZAWI, CHRISTOPHE BRUTTIN, AND AMR HUBER. **Offline and Online Evaluation of News Recommender Systems at Swissinfo.Ch**. In *Proceedings of the 8th ACM*

Conference on Recommender Systems, RecSys '14, pages 169–176, New York, NY, USA, 2014. ACM. (34)

[25] MOUZHI GE, CARLA DELGADO-BATTENFELD, AND DIETMAR JANNACH. **Beyond accuracy: evaluating recommender systems by coverage and serendipity**. In *In RecSys 10*, page 257, 2010. (38)

[26] KEN GOLDBERG, THERESA ROEDER, DHRUV GUPTA, AND CHRIS PERKINS. **Eigentaste: A Constant Time Collaborative Filtering Algorithm**. *Inf. Retr.*, **4**(2):133–151, July 2001. (13)

[27] J. A. HARTIGAN. **Direct Clustering of a Data Matrix**. *Journal of the American Statistical Association*, **67**(337):123–129, 1972. (89, 90)

[28] TROND HELLEM, BJARTE DYSVIK, AND INGE JONASSEN. **LSimpute: accurate estimation of missing values in microarray data with least squares methods.** *Nucleic acids research*, **32**(3):e34+, February 2004. (60, 61)

[29] JONATHAN L. HERLOCKER, JOSEPH A. KONSTAN, AL BORCHERS, AND JOHN RIEDL. **An Algorithmic Framework for Performing Collaborative Filtering**. SIGIR '99, pages 230–237, New York, NY, USA, 1999. ACM. (17, 35)

[30] THOMAS HOFMANN. **Latent Semantic Models for Collaborative Filtering**. *ACM Trans. Inf. Syst.*, **22**(1):89–115, January 2004. (13)

[31] JOHN H. HOLLAND. *Genetic Algorithms and Adaptation*, pages 317–333. Springer US, Boston, MA, 1984. (22, 25)

[32] KATSUHIRO HONDA. **Fuzzy Co-Clustering and Application to CF**. In *Integrated Uncertainty in Knowledge Modelling and Decision Making*, pages 16–23. Springer Intl Publishing, 2016. (90)

[33] KE JI, ZHENXIANG CHEN, RUNYUAN SUN, KUN MA, ZHONGJIE YUAN, AND GUANDONG XU. **GIST: A generative model with individual and subgroup-based topics for group recommendation**. *Expert Systems with Applications*, **94**:81 – 93, 2018. (20)

[34] RONG JIN, JOYCE Y. CHAI, AND LUO SI. **An Automatic Weighting Scheme for Collaborative Filtering**. SIGIR '04, pages 337–344, New York, NY, USA, 2004. ACM. (17)

[35] MARIUS KAMINSKAS AND DEREK BRIDGE. **Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems**. *ACM Trans. Interact. Intell. Syst.*, **7**(1):2:1–2:42, December 2016. (38)

[36] SURYA KANT AND TRIPTI MAHARA. **Nearest biclusters collaborative filtering framework with fusion**. *Journal of Computational Science*, **25**:204 – 212, 2018. (2, 19, 20, 90)

[37] DERVIS KARABOGA. **An idea based on honey bee swarm for numerical optimization**. Technical report, 2005. (22)

[38] J. KENNEDY AND R. EBERHART. **Particle swarm optimization**. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, **4**, pages 1942–1948 vol.4, Nov 1995. (22, 23)

[39] HYUNSOO KIM, GENE H. GOLUB, AND HAESUN PARK. **Missing value estimation for DNA microarray gene expression data: local least squares imputation**. *Bioinformatics*, **21**(2):187–198, 2005. (60, 61, 62, 63)

[40] BART P. KNIJNENBURG, MARTIJN C. WILLEMSEN, ZENO GANTNER, HAKAN SONCU, AND CHRIS NEWELL. **Explaining the User Experience of Recommender Systems**. *User Modeling and User-Adapted Interaction*, **22**(4-5):441–504, October 2012. (33)

[41] RON KOHAVI, ROGER LONGBOTHAM, DAN SOMMERFIELD, AND RANDAL M. HENNE. **Controlled experiments on the web: survey and practical guide**. *Data Min. Knowl. Discov.*, **18**(1):140–181, 2009. (34)

[42] HAMIDREZA KOOHI AND KOUROSH KIANI. **User based CF using FCM**. *Measurement*, **91**:134 – 139, 2016. (90)

[43] YEHUDA KOREN, ROBERT BELL, AND CHRIS VOLINSKY. **Matrix factorization techniques for recommender systems**. *Computer*, **42**(8), 2009. (1, 4, 13, 29)

[44] DENIS KOTKOV, SHUAIQIANG WANG, AND JARI VEIJALAINEN. **A Survey of Serendipity in Recommender Systems**. *Know.-Based Syst.*, **111**(C):180–192, November 2016. (38)

[45] DANIEL LEE AND H SEBASTIAN SEUNG. **Learning the Parts of Objects by Non-Negative Matrix Factorization**. *Nature*, **401**:788–91, 11 1999. (13, 17)

[46] G. LINDEN, B. SMITH, AND J. YORK. **Amazon.com recommendation: item-to-item collaborative filtering**. *IEEE Internet Computing*. (1, 13, 17)

[47] DARIUSH ZANDI NAVGARAN, PARHAM MORADI, AND FARDIN AKHLAGHIAN. **Evolutionary based matrix factorization method for collaborative filtering systems**. *Electrical Engineering (ICEE), 2013 21st Iranian Conference on*, pages 1–5, 2013. (3, 7, 21, 28)

[48] NING LI, YUAN-QING QIN, DE-BAO SUN, AND TONG ZOU. **Particle swarm optimization with mutation operator**. In *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*, **4**, pages 2251–2256 vol.4, Aug 2004. (7)

[49] MICHAEL J. PAZZANI AND DANIEL BILLSUS. **The Adaptive Web**. chapter Content-based Recommendation Systems, pages 325–341. 2007. (1)

[50] PEARL PU, LI CHEN, AND RONG HU. **A User-centric Evaluation Framework for Recommender Systems**. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 157–164, New York, NY, USA, 2011. ACM. (33)

[51] ZHI QIAO, PENG ZHANG, YANAN CAO, CHUAN ZHOU, AND LI GUO. **Improving Collaborative Recommendation via Location-based User-item Subgroup**. *Procedia Computer Science*, **29**(Supplement C):400 – 409, 2014. 2014 International Conference on Computational Science. (20)

[52] JASSON D. M. RENNIE AND NATHAN SREBRO. **Fast Maximum Margin Matrix Factorization for Collaborative Prediction**. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 713–719, New York, NY, USA, 2005. ACM. (13, 79, 84, 85, 97)

[53] RUSLAN SALAKHUTDINOV AND ANDRIY MNIH. **Probabilistic Matrix Factorization**. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, pages 1257–1264, USA, 2007. Curran Associates Inc. (13, 97)

[54] B. M. SARWAR, G. KARYPIS, J. A. KONSTAN, AND J. T. RIEDL. **Application of Dimensionality Reduction in Recommender Systems: A case study**. In *WebKDD Workshop at the ACM SIGKKD*, 2000. (13)

[55] BADRUL SARWAR, GEORGE KARYPIS, JOSEPH KONSTAN, AND JOHN RIEDL. **Item-based collaborative filtering recommendation algorithms**. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001. (xiv, 17, 18, 79, 80, 83, 84, 85, 97)

[56] BADRUL M. SARWAR, GEORGE KARYPIS, JOSEPH KONSTAN, AND JOHN RIEDL. **Recommender Systems for Large-scale E-Commerce: Scalable Neighborhood Formation Using Clustering**. In *International Conference on Computer and Information Technology*, 2002. (13)

[57] C. SELVI AND E. SIVASANKAR. **A novel optimization algorithm for recommender system using modified fuzzy c-means clustering approach**. *Soft Computing*, **23**(6):1901–1916, Mar 2019. (3, 28)

[58] GUY SHANI AND ASELA GUNAWARDANA. **Evaluating Recommendation Systems**. In *Recommender Systems Handbook*, pages 257–297. 2011. (4, 31, 35, 38, 39)

[59] MONIKA SINGH AND MONICA MEHROTRA. **Impact of biclustering on the performance of Biclustering based Collaborative Filtering**. *Expert Systems with Applications*, **113**:443 – 456, 2018. (2, 20)

[60] NATHAN SREBRO, JASON RENNIE, AND TOMMI S JAAKKOLA. **Maximum-margin matrix factorization**. In *Advances in neural information processing systems*, pages 1329–1336, 2004. (2, 13, 14, 15)

[61] PANAGIOTIS SYMEONIDIS AND ET. AL. **Nearest-biclusters Collaborative Filtering Based on Constant and Coherent Values**. *Inf. Retr.*, **11**(1):51–75, February 2008. (19, 20, 90)

[62] PANAGIOTIS SYMEONIDIS, ROS NANOPOULOS, APOSTOLOS PAPADOPOULOS, AND YANNIS MANOLOPOULOS. **Nearest-Biclusters Collaborative Filtering with Constant Values**. (20)

[63] OLGA G. TROYANSKAYA, MICHAEL N. CANTOR, GAVIN SHERLOCK, PATRICK O. BROWN, TREVOR HASTIE, ROBERT TIBSHIRANI, DAVID BOTSTEIN, AND RUSS B. ALTMAN. **Missing value estimation methods for DNA microarrays**. *Bioinformatics*, **17**(6):520–525, 2001. (60)

[64] SUPIYA UJJIN AND PETER J BENTLEY. **Particle swarm optimization recommender system**. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pages 124–131. IEEE, 2003. (2, 28)

[65] LYLE H UNGAR AND DEAN P FOSTER. **Clustering methods for collaborative filtering**. In *AAAI workshop on recommendation systems*, **1**, pages 114–129, 1998. (13)

[66] MOHAMMED WASID AND VIBHOR KANT. **A Particle Swarm Approach to Collaborative Filtering based Recommender Systems through Fuzzy Features**. *Procedia Computer Science*, **54**:440 – 448, 2015. (2, 21, 28)

[67] M. WU. **Collaborative Filtering via Ensembles of Matrix Factorizations**. In *KDD Cup and Workshop 2007*, pages 43–47. Max-Planck-Gesellschaft, August 2007. (13, 15)

[68] BIN XU, JIAJUN BU, CHUN CHEN, AND DENG CAI. **An exploration of improving collaborative recommender systems via user-item subgroups**. In *Proceedings of the 21st international conference on World Wide Web*, pages 21–30. ACM, 2012. (2, 13, 19, 20, 37, 59, 79, 80, 84, 85, 90, 91, 92, 97, 99)

[69] GUI-RONG XUE, CHENXI LIN, QIANG YANG, WENSI XI, HUA-JUN ZENG, YONG YU, AND ZHENG CHEN. **Scalable Collaborative Filtering Using Cluster-based Smoothing**. SIGIR '05, pages 114–121, New York, NY, USA, 2005. ACM. (13, 17)

[70] YE YANG AND YUNHUA ZHANG. **CF recommendation model based on fuzzy clustering algorithm**, 05 2018. (90)

Metaheuristics for collaborative filtering in recommender systems

by Ayangleima Laishram

Metaheuristics for collaborative filtering in recommender systems

ORIGINALITY REPORT Overall Similarity Index : 38-28 = 10%

29/6/2024
Professor
School of CIS
Prof. C.R. Rao Road,
Central University
Hyderabad-46. (India)

37%	7%	36%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Ayangleima Laishram, Vineet Padmanabhan. "Discovery of user-item subgroups via genetic algorithm for effective prediction of ratings in collaborative filtering", Applied Intelligence, 2019 Publication This Publication is by the Student	18%
2	"Advances in Artificial Intelligence", Springer Science and Business Media LLC, 2019 Publication This Publication is by the Student	9%
3	mafiadoc.com Internet Source	3%
4	Ayangleima Laishram, Vineet Padmanabhan, Rajendra Prasad Lal. "Analysis of similarity measures in user-item subgroup based collaborative filtering via genetic algorithm", International Journal of Information Technology, 2018 Publication This Publication is by the Student.	1%
5	Ayangleima Laishram, Vineet Padmanabhan. "Chapter 8 Enhanced Collaborative Filtering	<1%