UNIVERSIDAD DE GRANADA

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Novel Models in Recommender Systems and Group Recommender Systems for Improving Recommendations

Programa Oficial de Doctorado en Tecnologías de la Información y la Comunicación

Tesis Doctoral

Jorge Castro Gallardo

Granada, Noviembre de 2018

Editor: Universidad de Granada. Tesis Doctorales
Autor: Jorge Castro Gallardo
ISBN: 978-84-1306-139-9
URI: http://hdl.handle.net/10481/55453

Esta tesis doctoral ha sido desarrollada con la financiación de una beca predoctoral FPU (código FPU13/01151) adscrita al Ministerio de Educación y Ciencia. También ha sido subvencionada por los proyectos TIN2012-31263, TIN2015-66524-P del Ministerio de Ciencia e Innovación.

Agradecimientos

Dado que trabajo del investigador requiere un esfuerzo y trabajo cercano a lo sobrehumano, quiero dedicar unas líneas a las personas que me han acompañado de una manera u otra en el trabajo de estos últimos años, que me han apoyado y sin las que no habría podido llegar hasta aquí.

Entre todas ellas destaco a mi familia, mis padres, Ángel y María Dolores, sin su educación y disciplina no hubiera podido aspirar a tan ardua tarea. A mi pareja, Isabel, por su infinito apoyo, paciencia y compañía durante esta fase de mi etapa investigadora. A mi hermano, Ángel Darío, por todos los momentos compartidos. Al resto de familiares y amigos que han contribuido de alguna manera en mi etapa doctoral.

En el mundo académico quiero agradecer a mi director, Luis Martínez López, su esfuerzo, paciencia y confianza durante estos años sin los cuales no hubiera podido llegar a alcanzar los excelentes resultados que ahora presento. También quiero agradecer a los profesionales académicos que han ofrecido apoyo y formación durante estos años, como Oscar Cordón, Paco Herrera, Salvador García, Rosa M. Rodríguez, Manuel J. Barranco, Macarena Espinilla y Pedro Sánchez, entre otros.

También en el mundo académico quiero agradecer a mis compañeros durante mi etapa como aspirante al grado de doctor, como Manu Parra, Rosa Rodríguez, Sara, Pablo, Dani Peralta, Manuel Titos, Lala, Jose, Antonio, Edu, Fran, Juanan, Rafa, Sergio Gonzalez, Jesús Maillo, Elena, Jose Ángel, Eva, Nacho, Andrea, Salva, Julián Luengo, Eugenio, Francisco J. Estrella, Francisco J. Quesada, Raciel Yera, Yeleny Zulueta, Hong Bing, Jun Liu, Shuwei Chen y Francisco Moya, entre otros tantos más. Gracias a todos.

Special thanks to the colleagues of the DeSI group during my stay at the University of Technology Sydney, with a special mention to Jie Lu, Guanguan Zhang, Hongshu Chen, Chenlian Hu and Brenda Wang, for the treatment and hospitality received.

Y por último pero no por ello menos importante, agradezco a mis amigos Sergio, Ana, Javi y Raúl, entre otros, por su apoyo en los momentos de desconexión del trabajo.

Muchas gracias a todos.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Currently, the growth of online information available in services generates that users are not able to search and find items that suit their preferences and needs due to the large amount of alternatives to explore. This problem is known as *information overload*. Personalization techniques try to overcome this problem providing the users with personalized access to information tailored to their preferences and needs. Recommender systems [45] are the most successful personalization tools that filter relevant alternatives to select the most promising ones regarding users interest. Several advances have been made to increase the performance of recommender systems, such as their improvement in cold-start scenarios [131, 139, 249], scalability [158] or extension to new domains [137], among others. Researchers have identified new research topics such as group recommendation [114, 144], or to integrate new sources of information, such as context-aware recommender systems [4, 6] or social network recommendation [87], among others.

The research on group recommender systems is motivated by the existence of social products that are usually consumed by groups of users [114, 144]. Therefore, recommendations need to be tailored to a group of users that may have different or even conflicting interests [37, 228]. Hence, individual recommender systems are often extended for group recommendation through the aggregation of preferences or recommendations. This aggregation aims at generating a collective rec-

ommendation from individuals' information. Moreover, the extension of individual recommender systems through aggregation makes individual models available for group recommendation [144]. However, this extension is not often straightforward or it has some limitations.

A limitation of aggregation-based group recommender systems is that the first step aggregates individual ratings to build a pseudo-user that represents the group profile [144]. However, this aggregation results in information loss, which affects the shape and diversity of the ratings. Moreover, this aggregation does not consider that users' behavior change when they become part of a group, and they often negotiate their preferences to make them closer to the collective opinion and reach agreed solutions. In addition, there are also group dynamics that aggregation-based group recommender systems overlook, such as the relationships between members' preferences. Moreover, researchers have pointed out that ratings also present inconsistencies and proposed some techniques to mitigate its effect in individual recommender systems, but group recommendation lacks of studies in this direction. These current limitations of group recommender systems motivate the need for novel models that provide these features.

It has been also pointed out the utility of the users' context to improve recommendation quality [8, 173, 223]. Context-aware recommender systems aim to consider the specific user's context to adjust the recommendation to it. Therefore, context-aware recommender systems assume that recommendation quality depends not only on users' preferences, but also on the context in which the user receives the recommendation [8]. Context-aware recommender systems motivation is to improve recommendations. Content-based and collaborative filtering are the most widespread approaches. Hence, this research focuses on the study of context-awareness in both.

Context-awareness was not considered in early recommender systems. The availability of users' contextual information, particularly through sensors in mobile devices, motivated the proposal of context-aware recommender systems [8,

183]. Although there are several recommendation models that integrate context-awareness, few of them combine group recommendation and context-awareness. This issue is particularly important in tourism recommender systems, where context-awareness [222] and group recommendation [94] have been pointed out as interesting features. Therefore, research in this direction is needed to provide models fulfilling these requirements.

1.2 Research questions

This research is devoted to the development of new recommendation and group recommendation approaches and the integration of contextual features in them. To summarize, the following research questions will be answered by this research:

Research Question 1. How to mitigate the information loss implied in the aggregation step for the improvement of group recommender systems based on collaborative filtering?

Research Question 2. How to improve the recommendation in group recommender systems with consensus reaching processes?

Research Question 3. How to model the influences among members' preferences to improve group recommendation?

Research Question 4. How to extend natural noise management techniques to reduce users inconsistencies to improve the results of a group recommender system?

Research Question 5. How to effectively integrate contextual information extracted from microblogging services in question answering recommender systems?

Research Question 6. How to extend tourism recommender systems with context-awareness and group recommendation for the improvement of recommendation utility?

1.3 Research objectives

This research aims to achieve the following objectives to answer the above research questions:

Research Objective 1. To develop a hesitant fuzzy sets-based representation of group's preferences.

This research objective corresponds to Research Question 1. A method to represent group's preferences using hesitant fuzzy sets will be proposed. Here, the multiple ratings stated over one item by all members will be modeled as the group hesitation regarding the rating of such an item.

Research Objective 2. To develop a hesitant fuzzy sets-based group recommendation approach.

This research objective corresponds to Research Question 1. A novel recommendation method called HGRM will be developed. In this method, the group's preference modeling with hesitant fuzzy sets will be used to find suitable neighbors to the target group. The hesitant Pearson's correlation coefficient will allow to compute similarities between groups and users to build the group neighborhood. After that, a prediction for the group will be calculated using their neighborhood.

Research Objective 3. To develop a group recommendation approach based on the minimum cost consensus model.

This research objective corresponds to Research Question 2. A group recommendation approach based on the minimum cost consensus model named MinCostGRS will be developed. This group recommendation approach will apply the minimum cost consensus model to perform the minimum modifications to the individual preferences needed to achieve the desired consensus degree. After these modifications are calculated, the collective preference will be used to calculate the group recommendation with a high consensus, which improves group recommendation quality.

Research Objective 4. To develop a group recommendation approach with the automatic consensus support system model based on feedback.

This research objective corresponds to Research Question 2. A group recommendation approach based on consensus reaching processes named Consensus-GRS will be developed. This group recommendation approach will apply the automatic consensus support system model based on feedback to improve members agreement before the recommendation. This behavior will be achieved through the automatic negotiation and modification of individual preferences. After this automatic process finishes, the group recommendation will be calculated and the higher agreement between members will improve group recommendation quality.

Research Objective 5. To develop an extension for group recommender systems dealing with relationships among members with opinion dynamics model.

This research objective corresponds to Research Question 3. A framework to extend group recommender systems with opinion dynamics models will be developed. A model with the DeGroot's opinion dynamics model for group recommendation named Pre-GROD will be developed. The DeGroot's model will be used to reproduce the opinion change within the group regarding the relationships among members' preferences.

Research Objective 6. To develop a group recommendation approach that ensures consensus on members opinions within opinion dynamics based-group recommendation.

This research objective corresponds to Research Question 3. A group recommendation approach that uses such an extension of the DeGroot's model for group recommendation named GROD will be developed. Given that the opinion evolution in the DeGroot's model might not lead to consensus opinions, a modification of members' relationships will be developed to ensure that opinions always reach a consensus value.

Research Objective 7. To develop extensions of natural noise management focused on group recommender systems rating datasets.

This research objective corresponds to Research Question 4. Three natural noise management techniques for group recommender systems named NNM-LL,

NNM-LG, and NNM-GG will be developed. The specific features of the group recommendation problem reveal two levels in the ratings dataset: the local level regarding the group and the global level. The techniques developed will consider these aspects to detect and correct noisy ratings within group recommender systems rating datasets.

Research Objective 8. To develop a pre-processing approach for natural noise management in group recommendation.

This research objective corresponds to Research Question 4. A natural noise management approach for group recommender systems ratings datasets named NNM-H will be developed. This method will hybridize the techniques proposed in the previous research objective. First, a global natural noise management process will be performed, which reduces the noise in the entire dataset. After that, a further step of refinement will be applied over the target group ratings. The hybridization of these two steps of natural noise management will provide improvements to group recommendation approaches that use the de-noised ratings dataset.

Research Objective 9. To develop a method for topic identification within collaborative trend interest.

This research objective corresponds to Research Question 5. A method for detecting topics within the collaborative trend interest extracted from microblogging systems will be developed. The status updates extracted from microblogging systems mix several topics. The method proposed will apply the fuzzy c-means clustering algorithm to detect the various topics in these status updates to characterize the collaborative trend interest.

Research Objective 10. To develop a context-aware question answering recommendation approach considering collaborative trend interest.

This research objective corresponds to Research Question 5. A method for semantic context-aware question answering recommendation named LSAContextCluster will be developed. This approach will use the method proposed in the previous research objective. The collaborative trend interest characterisation will be con-

sidered as the user's context, which will allow to contextualize users' preference profiles. The recommendation for a target user will be calculated using the contextualized user's preference profile, which will provide both context-aware and personalized recommendations.

Research Objective 11. To develop a prototype for location and trajectory-aware recommendation.

This research objective corresponds to Research Question 6. A method to provide location and trajectory-aware recommendations will be developed. The traditional recommendation model based on user's preferences will be extended to integrate location and trajectory-aware recommendations. The resulting model will be integrated in a working prototype built upon the REJA recommender system.

Research Objective 12. To develop a prototype for consensus-driven context-aware group recommendation.

This research objective corresponds to Research Question 6. A method to provide consensus-driven context-aware recommendations will be developed. The consensus GRS model will be extended to integrate location-aware recommendations. The resulting model will be integrated in a working prototype built upon the REJA recommender system.

1.4 Research significance

This research work has both theoretical and practical significances in the area of recommender systems.

1.4.1 Theoretical significance

Theoretically, the research proposes several recommendation approaches that cover the following aspects of recommender systems:

- **Profile modeling.** The research carried out solves the problem of representing group's preferences effectively, which arises from Research Objective 1. The representation used applies hesitant fuzzy sets concepts, which allows

to model the multiple ratings of group's members as the group hesitation towards the rating of a given item.

- **Neighborhood identification.** This research solves the problem of calculating the neighborhood of a group of users without requiring a rating aggregation step that builds a pseudo-user. This problem arises from Research Objective 2. It is important to remark that the aggregation process implies a loss of information that is avoided in the neighborhood formation step with such a suitable group's preference modeling. A novel collaborative filtering-based group recommendation approach is proposed using the hesitant Pearson's correlation coefficient to achieve more accurate and diverse recommendations for groups of users.

- **Consensus recommendations.** This research solves the problem of providing agreed recommendations to groups of users that express their preferences individually. This problem arises in Research Objective 3, Research Objective 4 and Research Objective 6. Three methods that provide consensus recommendation are developed in this research. The MinCostGRS approach applies the minimum cost consensus model to find the modification of users' preferences that leads to high consensus with the minimum amount of changes. The ConsensusGRS approach applies the automatic consensus support system model based on feedback, which identifies group's members whose opinions are far and updates their preferences to make them closer to each other. The GROD approach considers members' opinions evolution guided by the relationships between members' preferences and the consensus recommendation is ensured adding relationships between opinion subgroups.

- **Group dynamics.** This research solves the problem of considering group dynamics in the group recommendation process, which arises from Research Objective 5. A novel group recommendation approach based on opinion

dynamics simulates the opinion evolution within a group of users regarding the relationships between their preferences.

- **Natural noise.** This research solves the natural noise management problem in group recommender systems, which arises from Research Objective 7. The NNM-LL, NNM-LG, NNM-GG and NNM-H approaches provide several alternatives for managing the natural noise in group recommender systems applying various assumptions regarding the ratings that are used for the noise identification and the ratings that are analyzed to be later corrected if they are identified as noisy. There was no previous approach focused on the management of natural noise in group recommendation.

- **Contextual modeling.** This research solves the problem of characterizing the context to include it in recommender systems. Referring to Research Objective 9, the fuzzy c-means clustering algorithm is used to identify the various topics that are present in the collaborative trend interest extracted from microblogging services. Thus, recommender systems can use this contextual modeling to provide context-aware recommendations.

1.4.2 Practical significance

The research carried out provides various ways to improve recommender systems performance regarding various recommendation quality viewpoints:

Accuracy. A major concern in recommender systems is to improve recommendation accuracy to better satisfy individual preferences. The models developed in this research provide better recommendations regarding their accuracy.

Diversity. The diversity of the recommendations has been identified as a positive aspect in recommender systems. HGRM developed in this research improves recommendation diversity while it maintains an acceptable accuracy. This property is interesting to reduce the chance of a member not satisfied with any recommended item.

Efficiency. The volume of information produced by users in recommender system makes the efficiency of recommendation methods an important issue. The proposed methods for group recommendation and noise management in group recommendation are designed to have a computational complexity order similar to the method that they improve, which makes the investment on additional computation resources worth because they improve the quality of recommendations regarding the aspects considered.

1.5 Research methodology and process

The overall research methodology and research process are designed as follows.

1.5.1 Research methodology

Research methodology is the "collection of problem solving methods governed by a set of principles and a common philosophy for solving targeted problems" [84]. This research is done in information systems domain. Several research methodologies have been proposed and applied to the information systems domain, such as case study, field study, design research, archival research, field experiment, laboratory experiment, survey and action research [160, 207, 220].

1.5.2 Research process

The research plan was developed according to the methodology of design research. First, recommender systems research area was chosen as the broad research topic of this research. The recommender systems area was analyzed performing a literature review of previous research, where existing literature was retrieved and critically reviewed. The specific research questions addressed in this research were identified from previous results of the literature review. A refinement process was performed to clearly define the precise formulation of the research questions reviewing more literature closely related to the initial research questions. The key research gaps identified in the literature include modeling group's preferences, neighbor selec-

tion in group recommendation, group dynamics, consensus in group recommendation, natural noise and contextual modeling. From the objectives set, this research presents novel models for group recommendation and context-aware recommendation. Datasets publicly available for researchers in these domains are used to evaluate the proposed recommendation approaches. The results of the experiments done to evaluate the novel proposals allowed to further refine the novel proposals in an iterative process until the results obtained were satisfactory. Finally, the PhD thesis was written at the end of the research.

1.6 Thesis structure

This thesis contains nine chapters. Chapter 1 presents the research motivation, research questions, objectives, significance, research methodology and process. Chapter 2 presents the literature relevant to this study, which includes classical and state-of-the-art recommendation techniques and applications. Also, related concepts are introduced in this chapter. Chapter 3 proposes a novel group recommender system that avoids information loss using hesitant fuzzy sets concepts. Chapter 4 proposes a framework for consensus-driven group recommender systems that integrates consensus reaching processes from group decision making to the group recommendation problem. Chapter 5 develops a group recommender system based on opinion dynamics model to consider relationships among members' preferences and ensure consensus recommendations. Chapter 6 studies the extension of natural noise management models to the group recommendation problem and proposes an effective pre-processing method to reduce natural noise in group recommender systems datasets. Chapter 7 presents a method for recommending question answering items to individuals using the collaborative trend interest as context in the recommendation. Chapter 8 presents two prototypes built upon REJA to integrate location and trajectory-aware recommendations and location-aware consensus-driven group recommendations. Chapter 9 presents the conclusions and further study recommendations. The structure and content of the thesis

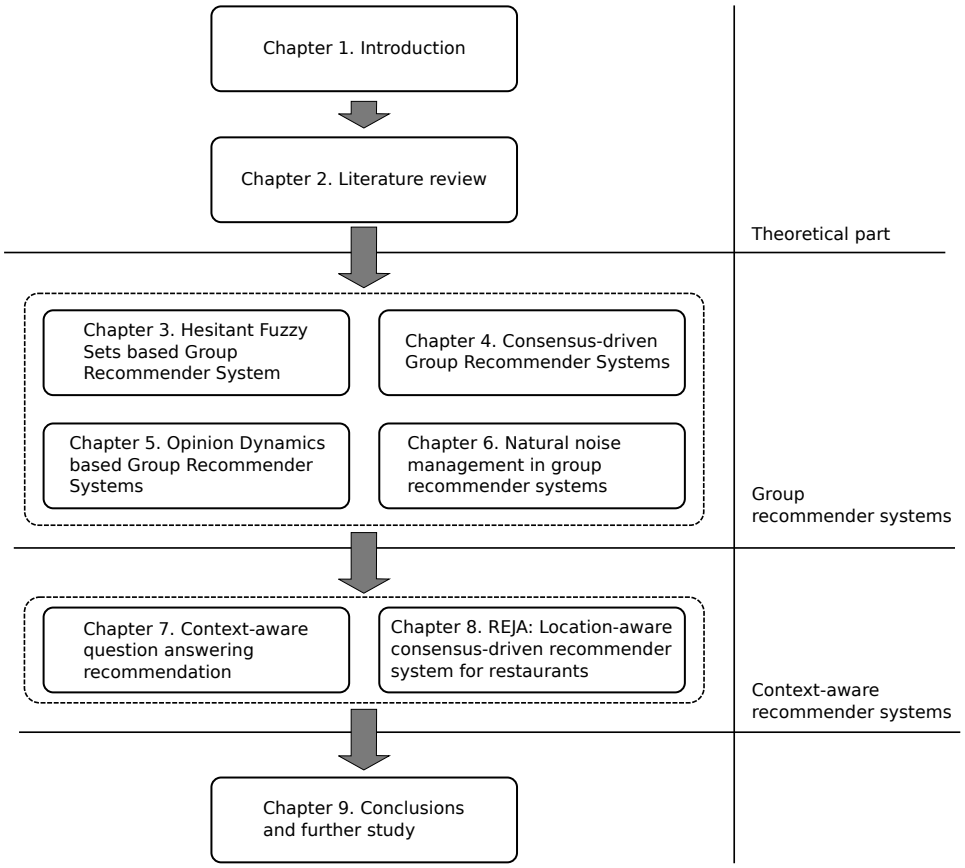


Figure 1.1: Structure and contents of this thesis.

are as indicated in Figure 1.1

1.7 Publications related to this thesis

- Jorge Castro, Manuel J. Barranco, Rosa M. Rodrǵuez and L. Martínez (2017), Group Recommendations Based on Hesitant Fuzzy Sets. International Journal of Intelligent Systems 33: 2058-2077. doi: 10.1002/int.21922

- Jorge Castro, Francisco J. Quesada, Iván Palomares, and Luis Martinez (2015). A Consensus-Driven Group Recommender System. International Journal of Intelligent Systems 30(8): 887-906. doi: 10.1002/int.21730.

- Jorge Castro, Jie Lu, Guangquan Zhang, Yucheng Dong, Luis Martínez

(2018). Opinion dynamics-based group recommender systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems 48 (12): 2394-2406. doi: 10.1109/TSMC.2017.2695158

- Jorge Castro, Raciel Yera, and Luis Martínez (2017). An empirical study of natural noise management in group recommendation systems." Decision Support Systems 94: 1-11. doi: 10.1016/j.dss.2016.09.020

- Jorge Castro, Jie Lu, Guangquan Zhang, Luis Martínez. A question answer-based context-aware recommendation method with semantic model within a big data framework. IEEE transactions on Cybernetics. (submitted)

- Jorge Castro, Oscar Cordón, Luis Martínez (2015). CLG-REJA: A consensus location-aware group recommender system for restaurants. Workshop on Tourism Recommender Systems, within RecSys'15, 20th September 2015, Vienna (Austria)

- Manuel J. Barranco, José M. Noguera, Jorge Castro, Luis Martínez (2012). A context-aware mobile recommender system based on location and trajectory. In Management Intelligent Systems: 153-162. doi: 10.1007/978-3-642-30864-2_15

Chapter 2

Literature Review

2.1 Introduction to background

Due to the easy access to information provided by Internet and the increasing amount of information available in World Wide Web scenarios, there is a rising number of situations, such as e-commerce, in which the number of alternatives available to users makes the selection of the most suitable one difficult. This situation originates that users need to put significant effort for finding relevant pieces of information. This difficulty, so called *information overload* [193], is often motivated by the increasing number of alternatives or by the amount of information available for each of them, which makes it not feasible for a user to revise all alternatives to select the most suitable one. Therefore, finding out items fitting users' preferences and needs effectively is an important challenge nowadays. Among the various approaches in the literature aimed to overcome information overload, personalization focuses on the analysis of user interests and needs to filter items that are relevant for the target user [152]. Recommender systems were proposed to filter information, thus delivering to users only the information that meets their preferences or needs. Recommender systems [192] have been a powerful personalization tool for alleviating information overload in large search spaces. Recommender systems have been proved to be successful in several domains, such as e-business [136], e-learning [230, 238], e-tourism [12, 161], e-commerce [187], web pages [157, 236] and financial investment [154], among others.

However, there are items with social features that are meant to be consumed by groups [143]. Examples of social items identified in the literature are movies [162], music [146], tourist points of interest [147, 161], tourist attractions [86] or television programmes [198], among others. In these situations, people form a group to consume the item, e.g., friends who want to choose a movie to watch, to eat at a restaurant, to select the vacation destination, etc. In these cases, the recommendations should satisfy the whole group instead of targeting individuals.

With this purpose in mind, group recommender systems help groups of users find suitable items according to their preferences and needs [9, 114]. Most approaches for group recommendation build upon individual recommender systems and provide the group recommendation aggregating individuals information [144]. A benefit of the aggregation-based extension is that individual preferences already gathered in individual systems can be used for groups. In the literature, two main approaches to aggregate individuals' information are identified. Some approaches aggregate individual ratings to form a collective profile that represents the group's preferences [167], while others aggregate individual recommendations to produce a single recommendation targeted to the group [18]. Within these aggregation approaches, the aggregation strategy used can also vary, such as average, least misery, or most pleasure, among others [144].

These aggregation processes might cause the overlook of distribution, diversity or shape of the initial data. This loss of information can either bias or lead to worse results and it may affect recommendation accuracy and diversity. Therefore, keeping the maximum information about the group during the recommendation process is an important challenge in group recommender systems to provide better recommendations. Therefore, a promising direction for research is to find a suitable preference modeling and information fusion tools that help to overcome such a limitation. Therefore, Hesitant Fuzzy Sets (HFS), an extension of Fuzzy Sets [240] introduced by Torra [217], concept can be applied. HFSs handle hesitation with multiple membership functions, which output several membership values, instead

of a single one. This feature can be used to model members' preferences as group hesitation towards the item rating. Operations defined over HFSs allow to compute the neighborhood of the group, which allows to extend collaborative filtering for group recommendation.

Previous researches highlighted the importance of delivering recommendations that do not leave any member dissatisfied. With regard to this issue, the selection of the group recommendation has been performed with the minimum aggregation [162], and items are selected regarding the satisfaction of the least satisfied member. Although this approach provides benefits, it does not consider that group's members try to find consensus solutions [78]. Consensus reaching process [109, 168] is a branch of group decision making [49, 135] that studies how individuals negotiate their preferences before selecting the best alternative in a decision problem. Such a process is done to maximize the acceptation of the final alternative, because members perceive that their preferences were considered in the selection of the final alternative and improve the agreement over the recommendation for the group. Therefore, the application of consensus reaching processes within group recommender systems is a promising direction of research to improve group recommendation. Hence, a framework for consensus-based group recommendation is proposed. Two group recommendation approaches that follow such a framework are proposed to integrate (i) the minimum cost consensus model, and (ii) the automatic consensus support system model based on feedback.

Moreover, aggregation-based group recommender sytems also overlook the dynamics that might exist in the group. Examples of these dynamics are the change of individual opinions when users belong to a group, or the influence among members due to trust relationships. With regard to this issue, opinion dynamics aims to model how opinion changes in a group of individuals [69]. There are various models regarding the rules that drive such an opinion change. Some models consider that opinion change is driven by closeness of opinions [102, 134], or random encounters [250], among others. Specifically, the application of models that con-

sider trust between people to drive the opinion change [68] allows to model how members' preferences influence each other. The consideration of this aspect is a promising direction for the improvement of recommendations for groups. A group recommender system based on opinion dynamics model is then developed.

Other reason for lower performance of recommender systems is the quality of user's preferences [30]. Previous researches pointed out that recommender systems could be reaching a *magic barrier* in their accuracy due to users inconsistencies when rating [103]. Among other reasons [165], rating quality is determined by the presence of natural noise, which can be defined as the inconsistencies in preferences introduced unmaliciously by users. The causes of natural noise can be differences in the rating process, influence of external factors, or human error [164]. Several approaches have been explored to manage the natural noise present in the preference elicitation process and minimise its negative impact in recommendation focused on recommender systems for individuals [15, 16]. The integration of natural noise management in the group recommendation scenario poses a new focus on users' preferences and the way they are treated in order to minimize natural noise influence on the recommendation. Therefore, the study of how can natural noise management be extended from the application in individual recommender system to group recommendation is an interesting direction for research. A proposal that considers the specific features of group recommendation is developed, which first manages the natural noise at the dataset level, and after that it further refines the management focusing on the target group ratings to improve the quality of the ratings provided to a group recommender system.

Previous recommender systems assume that user's satisfaction towards the recommendations is only dependent on user's preferences, thus finding the best item or set of items can be done analysing the ratings solely. But in some scenarios, the user's satisfaction with a given recommendation can also depend on other factors, such as the time when the recommendation is requested, the item's location, or the user's circumstances. Context-aware recommender systems (CARS) [6] are an

extension of traditional recommender systems that include contextual information in the recommendation calculation.

Researchers highlighted the inmediacy of microblogging systems [105]. This finding poses them as interesting sources of context. In microblogging systems, users can post a status update consisting of a free text input. The analysis of current trend interest within microblogging systems is a promising direction to improve recommendation with contextual features. Specifically, current trend interest influences the recommendation in the question answering domain. Within it, previous researches focused on finding the best answerers to reduce the answering time [208, 252]. This research focuses on the recommendation of already answered questions to improve users' knowledge. The recommendation in such a case is calculated from users' preferences, but it is also influenced by the context, which makes some questions more interesting than others. A semantic context-aware model for recommendation in the question answering domain is proposed. Within it, the characterization of context makes worth the application of clustering techniques with the aim of identifying the topics of the context. The user preference profile is combined with such contextual topic profiles to provide a contextualized user's profile, which can later be used to provide context-aware personalized recommendations in the question answering domain.

The integration of contextual information is particularly interesting in the tourism domain. In such a domain, the context greatly influences the satisfaction towards the recommendation, which makes some alternatives more interesting than others in certain contexts. Consequently, two prototypes for recommendation in the tourism domain are developed. These prototypes are built upon REJA, a recommender system of restaurants built in the University of Jaén that provides recommendations of restaurants in the province of Jaén. In these scenarios, the concurrence of both group recommendation and context-aware issues makes the development of models that consider both aspects necessary. Therefore, the recommender system of REJA is extended to integrate location and trajectory, and the consensus-

driven GRS model is extended to provide location-aware group recommendations through a contextual post-filtering process.

The remaining of this chapter presents a discussion of relevant works related to the research directions covered in this thesis. Section 2.2 reviews basic concepts, techniques and successful applications of recommender systems. Section 2.3 discusses the related works on group recommendation. Section 2.4 reviews context-aware recommender systems related works. Section 2.5 introduces concepts applied in the recommendation methods proposed in this research.

2.2 Recommender systems

In the last decades, there have been advances that provide an incredibly easy access to information. Users can search and find details of many elements in everyday life, such as restaurants, hotels, electronic devices, services, or events. When users need to make a decision about several alternatives, the vast amount of options available makes the evaluation of all options difficult. This problem is known as information overload.

To overcome the information overload problem, personalization techniques try to find interesting elements for users taking into account their preferences or history of interactions, among other features. Among existing personalization techniques, recommender systems are the most succesful ones [9, 122]. Specifically, recommender systems (RSs) [190] focus on personalizing the access that users have to the information filtering relevant elements [9, 75]. A good definition of a recommender system was given by R. Burke [45], and he defines it as *"any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options"*.

Currently, recommender systems are widely used in several domains to alleviate information overload, such as e-commerce, e-learning, e-government, tourism, web pages and digital games. Successful examples of applications are e-learning

[65, 138, 230, 238], e-business [136], e-commerce [32, 187, 234], e-tourism [12, 161, 205], e-government [99], financial investment [154], digital games [3], and web pages [53, 157, 236], among others.

Various approaches have been applied in recommender systems regarding the way they recommend and there are several taxonomies of recommender systems [45, 66]. Generally, recommendation techniques developed can be classified as collaborative filtering [126, 204], content-based [133, 177], knowledge-based [44], and hybrid techniques [45]. Among them, the most widespread ones are collaborative filtering (CF) [125] and content-based (CB) [63]. The main difference between them is that CF focuses on users' interaction with items, i.e., user's preferences, while CB focuses on the analysis of item's descriptions, i.e., item's content. Therefore, the performance of these recommendation approaches is subject to the quality and amount of available information of both types. In addition to these widespread strategies, other approaches have been proposed, such as knowledge-based recommender systems that focus on employing expert information over the recommendation domain through ontologies [158], among others [59, 230]. It is also worthy to remark the coexistence of two or more approaches in a recommender system through the hybridization, which aims to provide the benefit of the hybridized approaches or aim to overcome their limitations.

Formally, the recommendation problem can be formulated as a prediction problem [4] to find the most useful item, or set of most useful items, among a large set of choices. To find the best item, a prediction function is approximated by the recommender system:

$$Recommendation(I,u) = \arg\max_{i_k \in I}[Prediction(i_k,u)] \tag{2.1}$$

To obtain the recommendations, the recommender system may use information over the users ($U = \{u_1,\ldots,u_m\}$), the items ($I = \{i_1,\ldots,i_n\}$) and the users' ratings over a set of items ($R \subseteq U \times I \rightarrow D$), among other.

Depending on how the information is used to recommend, there are different

types of RSs:

- Collaborative filtering RS (CFRS) [124]. Among the different types of RSs, the most successful approach is CFRS, which analyzes users' preferences to recommend. This feature makes them able to recommend complex items, because they do not need any item knowledge to produce high quality recommendations.

- Content-based RS (CBRS) [133]. CBRSs rely on items' information, which can be a textual description or metadata (items' features) [54]. They also need users' feedback over items and they recommend items that are similar to the ones that the user already experienced and/or liked.

- Knowledge-based RS (KBRS) [219]. In KBRS, the system holds and uses any kind of additional knowledge, such as a user model created from some items that are given as an example of a good item [141], a tweak over the features of a given recommendation (critique-based), or domain specific knowledge that describes items' features and their relations (ontology-based)

The remaining of this section revises the successful techniques for recommendation, which comprise collaborative filtering, content-based filtering, knowledge-based recommendation and hybrid recommender systems.

Collaborative filtering

Collaborative filtering recommender systems are currently the most popular type of recommender systems in real world because of their simplicity and effectiveness. The researh in collaborative filtering began in the 1990s [91, 189, 209]. Initial collaborative filtering techniques are based on *word-of-mouth*, where the recommendations for the target user are calculated using similar users' preferences. CFRSs have the advantage of not needing information regarding the items because they treat the items as a black box and only use the preferences that users provided over

Table 2.1: Ratings table of a collaborative filtering recommender system

U	I				
	i_1	$\dots$	i_k	$\dots$	i_n
u_1	$r_{u_1 i_1}$	$\dots$	$r_{u_1 i_k}$	$\dots$	$r_{u_1 i_n}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
u_j	$r_{u_j i_1}$	$\dots$	$r_{u_j i_k}$	$\dots$	$r_{u_j i_n}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
u_m	$r_{u_m i_1}$	$\dots$	$r_{u_m i_k}$	$\dots$	$r_{u_m i_n}$

them to recommend. Therefore, the more users' preferences are available to the recommender system, the better performance.

The information that a typical collaborative filtering recommender system uses is depicted in Table 2.1. Here, the sets of users U provides ratings over the set of items I, where a given rating $r_{u_j i_k}$ notes the rating value that the user u_j gave to item i_k. It is worth to note that most of the ratings values are not known. The more ratings are unknown, the more sparsity the dataset has. Therefore, a typical user only gives ratings for a reduced subset of items.

One of the initial approaches proposed was the user-based collaborative filtering approach (UBCF) [91, 189]. In UBCF, the neighborhood of a user is computed first using a similarity measure to compare the preferences of a pair of users and the top-k are selected. After that, a prediction for each item not experienced yet by the target user is computed using the best neighbors' ratings over the target item. Finally, the items with the highest rating prediction are recommended to the target user. The UBCF method is also named memory-based collaborative filtering because the neighborhood is usually not stored and the computations are done in memory.

Pearson's correlation coefficient has been proved to be the most suitable similarity measure between users mostly because it is not affected by the user's bias when rating items [11], e.g., users who rate items on a consistently high or low basis. Once the neighborhood is selected, it is used to predict the rating for unseen

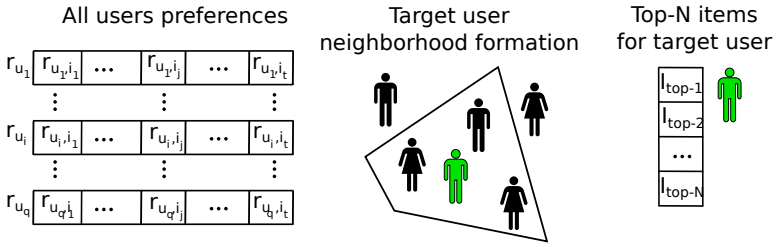


Figure 2.1: User-based collaborative filtering scheme.

items combining neighbors' ratings on the item. A number of methods to combine neighbors' ratings have been proposed [81]. One of the most used is the adjusted weighted sum, which considers the ratings of the neighbors with different weight regarding their similarity to the target user and also consider that users have a rating bias described by their average rating.

Other recommendation approach also based on the nearest neighbors algorithm was the item-based collaborative filtering approach (IBCF) [203]. In IBCF, a recommendation model is built, which contains for a given item the most similar items in the dataset. A prediction for a new item from the viewpoint of a target user u_k is computed using the target user's ratings over the neighbors of item i_k. The recommendation, similarly to UBCF, is composed of the items with the highest rating prediction.

These two techniques are widely spread. However, they suffer from problems associated to the availability of ratings. Cold-start problem [131, 249] is related to the absence of enough information regarding a given item or a given user, which makes that the recommender system is not able to compute recommendations or the recommentaion computed is of low quality. Other issue of collaborative filtering is the sparsity [5] of the ratings table. Usually, the recommendation quality with UBCF and IBCF decays when the sparsity of the ratings table is high.

Matrix factorization is the most successful approach among the model-based collaborative filtering approaches [126]. The idea behind matrix factorization is that user's preferences and item's features could be expressed as dimensions in

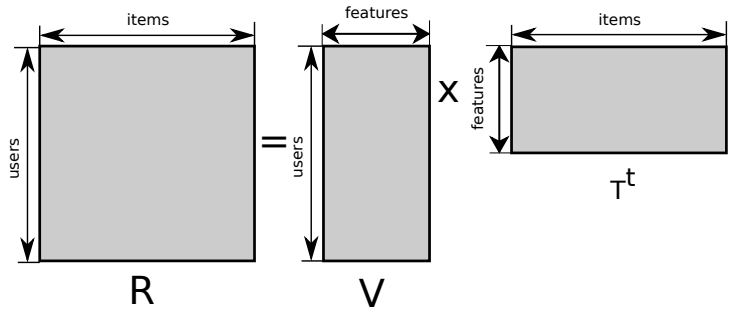


Figure 2.2: Scheme of the factorization of the rating matrix.

a set of hidden features. Therefore, a user's profile composed of many positive ratings for action movies would have a high value for the dimension for which action movies also have a high value. The process of extracting those dimensions and calculating user's and item's profiles is the factorization process of the rating matrix. The factorization problem can be expressed as stated in Equation 2.2 (see Figure 2.2.

$$R_{(|U|\times|I|)} = V_{(|U|\times f)} T^t_{(f\times|I|)} \tag{2.2}$$

where R is the rating matrix, V and T are low rank matrices whose multiplication approximates the original matrix R such that $VT^t = R' \approx R$, and f is the number of factors considered. Here, matrix V rows correspond to users' profiles, and T rows correspond to items' profiles. Hence, the recommendation with a matrix factorization model already computed is highly accurate and scalable. A rating prediction $\tilde{r}_{u_j i_k}$ with such a model consists of the multiplication of the corresponding feature vectors:

$$\tilde{r}_{u_j i_k} = v_{u_j} * t^t_{i_k} \tag{2.3}$$

Several approaches are applied to find suitable matrix factorization. Initial works applied singular value decomposition [202], but it relied on the rating matrix completion which biases the results in sparse datasets. Recent researches proposed several approaches to solve such a matrix factorization, such as stochastic gradient descent [83], or SVD++ [126], among others.

Content-based filtering

Content-based filtering [9, 36, 153, 177] is applied when items' descriptions are available in the dataset. CBRSs uses such descriptions, together with the available information about the choices that the user made in the past. The basic functions of a CBRS consist of (i) building content-based profiles from each item's description, (ii) building a preference profile for each user, (iii) comparing the available items' profiles with the user's profile and (iv) recommending the items that better fit the user's profile.

Within CBRSs two kinds can be distinguised: (i) based on item's features, and (ii) based on item's descriptions. This research focuses on the latter, given that QA items have a strong component of textual information for both formulating the question and answering it [79]. Although CBRSs suffer from user cold-start because they need some input from user's preferences and lack of diversity in recommendations [27], CBRS has demonstrated their utility when new items are introduced in the system, i.e., in scenarios with strong item cold-start [10]. This feature makes the application of CBRS approaches interesting in domains where new items are constantly introduced, such as web pages or news. Question answering recommendation shares the features to apply CBRS with textual descriptions, hence, this research focuses on this domain.

One of the first CBRS was proposed by Belkin and Croft [28] using information retrieval concepts, such as Rocchio's method. This system applies textual analysis to items' descriptions such that the features are the terms included in the item's description. With this procedure, a vector of ones and zeros that indicate which terms appear in the description is built to generate the item's profile.

The TFIDF approach [76] is also applied when item's content is given as item's description. In TFIDF, the free-text description is converted in structured data stemming words [184] to keep their root and reduce the number of components of documents. This is achieved because the stemmer unifies words such as computer, compute and computing, which are different forms that share meaning. A

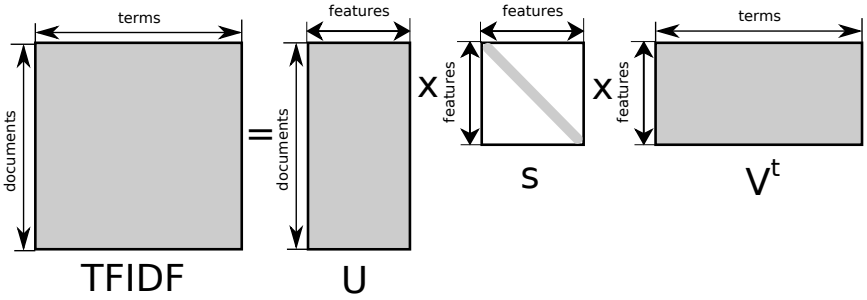


Figure 2.3: Decomposition of TFIDF matrix with singular Value decomposition. Note that s is a diagonal matrix with the singular values sorted in descending order.

vector of weights of each term is computed for each document multiplying $tf_{t,d}$ and idf_t [133]:

$$profile_d^{tfidf} = \{tf_{t,d} * idf_t \quad s.t. \quad t \in d\} \tag{2.4}$$

where $tf_{t,d}$ is the number of occurrences of term t in document d, N is the set of all documents and N_t is the set of documents that contain the term t at least once. The idf_t is defined as:

$$idf_t = -\log\left(\frac{|N|}{|N_t|}\right) \tag{2.5}$$

While TFIDF method is effective, it cannot deal with polisemy or synonym words. In order to overcome this issue, Latent Semantic Analysis (LSA) is applied [63]. In LSA, the term-document matrix is factorized with Singular Value Decomposition (SVD) to reduce it to orthogonal dimensions and keep the f most relevant singular values (see Figure 2.3).

$$TFIDF_{(|D|\times|T|)} = U_{(|D|\times f)} * s_{(f)} * V^t_{(f\times|W|)} \tag{2.6}$$

This way, a reduced feature space is defined, which properly manages noise and redundancy of terms. Users' profiles are generated from this feature-space definition through a linear combination of the profiles of documents that they liked [24]. Then, recommendations are generated comparing user's and documents' profiles using the cosine correlation coefficient.

Other methods are based on item's features [54, 215]. Such models assume that there are several item's features and each item can have a different value for each feature, e.g., in the case of a movie the features could be the leading actor, director, year of release and genre. In these cases, some approaches extended the TFIDF method to support feature-value descriptions [215]. Other research proposes the study of the relationships between feature values and the rating given to the item through entropy and dependency measures to allow a personalized weighting of the features for each user [54].

Content-based recommender systems do not suffer from item cold-start because a new item is readily available for recommendation. However, they suffer from user cold-start because a new user needs to provide sufficient information to the system in order to receive recommendations. Moreover, they also suffer from overspecialization because they can only recommend items that are similar to those already rated by the user. Therefore, content-based recommender systems cannot recommend items outside the target user's profile. In the case of feature-based content recommendation, the content-based approach cannot distinguish among items represented with the same feature values.

Knowledge-based recommender systems

Knowlege-based recommender systems use information specific to the recommendation domain, which is usually provided by experts or inferred from various available attributes. In these systems, users are often asked to explicitly provide the information neededed to complete their profile. Knowledge-based recommender systems are the most heterogeneous ones regarding the user interaction. In the literature, there are systems that request to the user only ratings and perform inferences from them [35], that request to the user items similar to those that the user is looking for [141], that allow the user to refine the recommendation list received [45, 46, 188, 224, 225, 185, 56], that receive as input preference relationships within a list of items [195], or that receive as input the prefererence relationship

among the item's features [42].

Knowledge-based recommender systems match users with items holding domain specific information about them. Usually, the system holds knowledge about how a given item meets users' preferences and needs. In order to do so, inferences about the properties of items and the needs of users might be done to recommend [45, 46].

A major drawback of knowledge-based recommender systems is that they require a great effort for building the initial knowledge database that they use. In return, the recommendations can be computed with a smaller amount of information from the user. Therefore, these systems are suitable to be used for a short term in initial phases of a system, or in situations in which other recommendation approaches are not able to provide accurate recommendtaions to users [43, 195].

Case-based reasoning recommender systems [211] use past problem solving experiences as a primary source to provide acceptable solutions for new problems [1], which is done in a four step process that consists of retrieve, reuse, revise and retain steps. Therefore, the system holds a database with the previous problems and the solutions provided. This way, the information stored can be used to solve a given new problem comparing its description to those of the already solved problems to select the most similar, and then the solution provided can be computed adjusting the solution of the similar solved problem. This behaviour is applied in recommender systems representing items as possible solutions and the user's profile as the problem specification [211]. Case-base recommender systems can be seen as a special type of content-based recommender systems, and they apply concepts from information retrieval for the case-based reasoning. However, case-based recommender systems differ from content-based ones because their item representations and item similarities are done in a different manner [211].

Other successful knowledge-based recommender systems are the constraint-based recommender systems [116, 77, 115, 245]. These systems allow an initial query that is refined incrementally to eliminate not satisfying products adding new

restrictions. The items recommended are those that satisfy all the constraints stated in the query. A limitation of constraint-based recommender systems is that certain sets of restrictions may not be satisfied by any item in the catalog, therefore, these systems orientate user when adding restrictions or detect and eliminate conflictive ones.

Hybrid recommender systems

A way of improving the recommendations is to hybridate two or more recommendation approaches. This hybridization is done with the aim of overcoming the limitations of the techniques used, or to provide the benefits of the techniques used. Burke [45] reviews the hybridizations used within recommender systems. In a later work, Zanker et al. [244] study these hybridization techniques and evaluate the hybridization of basic recommendation techniques regarding their accuracy. According to Burke [47] the following hybridization techniques are applied in recommender systems:

- Weighted: A numerical combination of the scores provided by each recommendation technque is performed to produce a single recommendation list [25, 80, 214].

- Switching: The recommendation approach is changed regarding certain criteria to use the best approach in each case [34, 90, 195, 214].

- Mixed: The recommendations from various recommendation approaches are presented together [212].

- Cascade: The items recommended by one recommendation approach are refined by the results of other recommendation approach [101, 111, 129].

- Feature combination: a prediction algorithm uses the output of several recommendation techniques and combines them [20, 64, 197].

- Feature augmentation: the output of a recommender system is used as a input feature of the following recommender system [148].

- Meta-level: The model outputed by one or several recommender systems is used as input of other recommender system [176, 243].

The most common aim when applying hybrid recommender systems is to overcome the cold-start, sparsity, or scalability problem of existing collaborative filtering recommender systems hybridizing them with other approaches [9, 29].

2.2.1 Research trends in recommender systems

Table 2.2 shows a comparison of the research in the recommender systems regarding the technique and the domain. As it can be seen, the majority of research is focused on the improvement of the traditional recommendation techniques mentioned along section 2.2: (i) collaborative filtering, (ii) content-based filtering, (iii) knowledge-based recommendation and (iv) hybrid recommendation models. Among the new research trends with few publications are (v) computational intelligence, (vi) social networks, (vii) context-aware, and (viii) group recommendation.

Due to our interest and research in group recommendation and in context-aware recommendation, this research focuses on both aspects. Specifically, regarding to group recommender systems, this research proposes several group recommendation models to overcome the limitations discussed in Section 2.1. On the other hand, regarding context-aware recommender system, this research extends previous models to provide context-awareness. The remainder of this chapter reviews both research lines.

2.3 Group recommender systems

There are certain items that are usually enjoyed in groups [144], thus traditional recommendation needs to be extended. Group recommender systems (GRSs) [114] were proposed to fill this gap. Specifically, this research focuses on GRSs that

Table 2.2: Publications in recommender systems according to the recommendation domain and technique applied.

	content-based	collaborative filtering	knowledge-based	hybrid	computational intelligence	social networks	context-aware	group recommendation
e-Government	1	5	1	5	4			
e-Business		1	3	3	4	1		
e-Commerce	3	1	4	1	4	2		
e-Library	2	2		3	1			
e-Learning	2		11		2			
e-Tourism	5	9	9	9	3	2	11	
e-Resource	9	16	6	15	8	1	1	
e-Group activity	9	5	2	5	1			2

use individual information to recommend, given that they benefit from individual usage.

An initial distinction that has to be done when studying a group recommender system is the actual element being recommended. This way, there are two kinds of GRSs: (i) GRS that recommend groups to users, and (ii) GRS that recommend items to groups.

- Recommend groups to users: These systems support users at finding relevant group activities. Also, they help activities organizers achieve greater participation. In this direction, MobiGroup, proposed by Guo et al. [97], is a system to help organize different group activities. Depending on the semantic of the activity (public, private), it applies a different approach to suggest people to involve in the activity. Additionally, MobiGroup takes into account the state of the activity, this is, if the activity is starting, running, or about to finish. Moreover, when the activity is private, the creator selects the people

to be informed about the activity. In the case of public activities, users are the ones that search for them and MobiGroup supports them filtering relevant ones. Users can also search running activities to join to, and therefore it takes into account both the user and the group context to show the best matching social activities. A study over MobiGroup shows that characterizing the diversity and regularity of group events leads to a better modeling of group activity recommendation. This is supported in an experiment in which the system is compared with a technique that does not take into account group characterization and context. The results prove that these features provide an increase of up to 30% on recommendation utility.

- Recommend items to groups: These systems are the focus of this research. In these systems, a group of users looks for a suitable item to enjoy together among a large amount of alternatives, therefore the task of the system is to find the item or items that best match the group interests and needs.

Moreover, GRSs are applied for different tasks, such as finding the most suitable group of users for a target item [251], or recommending groups to a user for joining them [155]. This research focuses on recommendation of items targeted to groups.

As pointed out by Borato et al. [37], there are different notions of group: (i) established groups, (ii) occasional groups, (iii) random groups, and (iv) automatically detected groups. This research focuses on (ii) occasional groups of people with certain aim in common, such as watching a movie together or going to a restaurant. It is important to consider the type of group that receives the recommendation in order to consider the best recommendation approach.

Group recommender systems (GRSs) compute recommendations targeted to groups [144] whose members can have different or even conflicting preferences [114]. The group recommendation problem can be formalized as follows:

$$Recommendation(G_a, I) = arg \max_{i_k \in I} Prediction(G_a, i_k) \tag{2.7}$$

where G is the target group, I is the set of available items and $Prediction(G_a, i_k)$ is a function that assigns a utility value for the item i_k regarding group G_a members. Table 2.3 summarizes the notation used in group recommender systems.

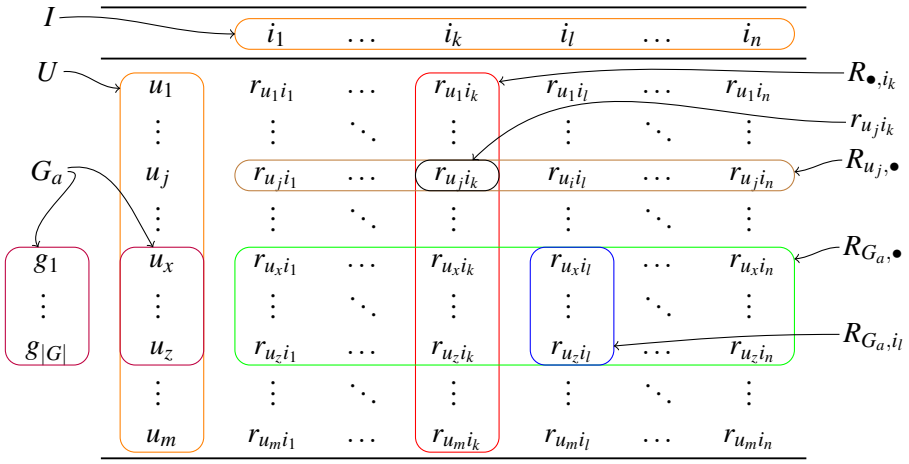


Table 2.3: Notation used group recommender systems to refer to ratings with their respective interpretation in terms of the set of elements that they refer to.

In group recommender systems, as stated by Jameson and Smyth [114], four basic recommendation subtasks are performed: (i) acquiring member preferences, (ii) generating recommendations, (iii) explaining group recommendations, and (iv) aiding to make the final choice.

The most widespread approach for group recommendation is to extend individual recommender systems [38, 51]. Therefore, the group recommendation problem is addressed reducing it to an individual recommendation problem by means of aggregating individual information. There are two aggregation approaches:

- Rating aggregation (see Figure 2.4): members state their preferences over some items. These ratings are aggregated to represent the group prefrence in a group profile named “pseudo-user”, which is then used by an individual recommender system [167].
- Recommendation aggregation (see Figure 2.5): From the individual prefereces, a recommender system computes the individual recommendations

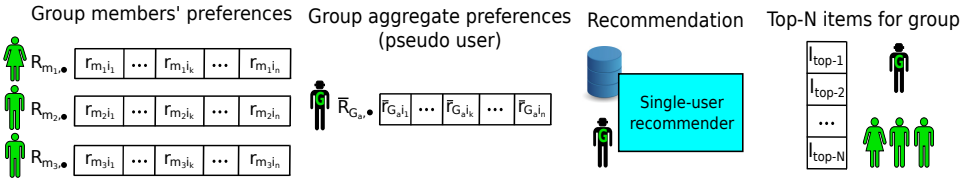


Figure 2.4: The framework of group recommender system based on rating aggregation.

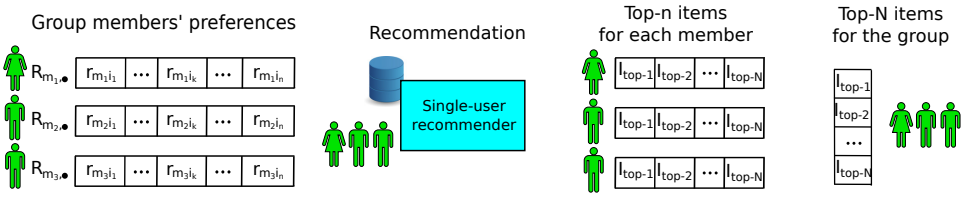


Figure 2.5: The framework of group recommender system based on recommendation aggregation.

for each member of the group. These individual recommendations are later combined to tailor the recommendations targeted to the group [18, 162].

Previous researches found that neither approach is better than the other in all scenarios [144, 167]. A study in each case is necesary to select the best approach. Moreover, these approaches rely on different aggregation strategies that can be also adjusted regarding the specific recommendation scenario [144]:

- Least misery: this aggregation tries to avoid member dissatisfaction with the recommended items. The group is as satisfied as the least satisfied member. Therefore, the group's preference for one item is the minimum individual preference.

- Average: the group's preference is the average of all the individual preferences.

- Average without misery: this aggregation averages individual ratings after excluding items with individual preferences below a certain threshold.

Each aggregation strategy provides different features to the process. The least misery strategy is suitable for small groups because when groups become larger,

there is a greater probability of an item having a negative rating, which might lead to a group profile composed of negative preferences. This behavior would bias the group recommendation [38]. Moreover, the least misery aggregation strategy is sensitive to new ratings because adding a new negative rating can change the group profile and modify the recommendation. On the other hand, average strategy considers all members' ratings, not just the low ones. In the case of needing a balance between considering low ratings and all ratings provided, the average without misery strategy aggregates ratings over items whose group rating is above certain threshold to avoid including least preferred items in the group profile. Consequently, hated items are avoided with this strategy.

Several researchers follow these approaches to implement a GRS. In this direction, Dooms et al. [71] proposed OMUS, a system to centralize and manage a household multimedia content, which might be distributed across several devices, such as hard drives, mobile devices, laptops or network attached storage. To solve this issue of scattered content, it provides a centralized content indexation, which makes multimedia content management easier for a household. To improve the capabilities of the system, it also provides personalized access to both individuals and group of users. This system personalizes the content regarding the group considering members relative importance and their presence when delivering the recommendations, which are computed with the hybridization of collaborative filtering and content-based recommendation.

Ortega et al. [167] focused on improving the core algorithm for group recommendation through the application of matrix factorization. This research explores the factorization of the user's profiles to extract the relevant features and evaluates different approaches for aggregating the factorized profiles. Experiments on the well known datasets MovieLens and Netflix with random users demonstrate that the approaches after factorization, before factorization, and weighted before factorization are good for small groups, large groups and for rapidly changing groups, respectively.

A common limitation of RSs is the recommendation to users with a small amount of information, i.e., cold-start. An interesting approach to improve the group recommendation in these situations is the usage of social networks. In this direction, Gartrell et al. [87] propose a system that takes into account social interactions, members expertise, and interest dissimilarity among group's members. The system is evaluated using a real-world user study. The experiment determines that the best results for each group is achieved with different algorithms, therefore more experiments should be carried out in order to determine the best approach.

People have different reasons to gather in groups such as family relations, homophily, or just because they live together. The approach in which groups make decisions differs from each other. Hence, a GRS that faces recommendation to a variety of groups must take this fact into account. Usually, when the system is targeted to groups of different nature, the recommendation approach is adapted to the different groups by hand: the system administrator checks the groups that are using the system and selects the recommendation model that best fits the data. The latter approach is time consuming and needs to be regularly revised to verify if the model is still well adapted to the data.

The model proposed by Guo et al. [98] uses personality features of the group's members. With this information, the system applies a social influence model to modify the preferences on the members. They perform individuals characterization based on personal attributes: personality, expertise, and susceptibility. They also take into account relationships between users: intimacy and interests similarity. They apply the Friedkin-Johnsen model [82] to consider these features and drive the social influence model. This approach results in a implicit characterization of individuals.

Guy et al. [100] evaluate how the recommendations can be calculated for online communities. They compare performance measuring share actions for different approaches to build a community profile taking inactive members, active members and owners of the community into account. They also analyze the size

and age of the communities. The variety of these communities leads to having different optimal approaches for each of them. They indicate that large communities perform better if inactive users are not considered in the profile. Similarly, old communities also perform better with this approach.

In the same way, Senot et al. [206] analyses different strategies for group recommendation aiming to determine the factors that influence the choice of an aggregation strategy. To do so, they perform an experiment to compare the best way of combining individual profiles. For such experiment, they build individual's and group's profiles from gathered users' behavior. They compare the aggregation approach that produces a group profile closer to the profile built from group behavior. This analysis results in a series of guidelines to select the aggregation approach for group recommendation.

Group recommendation aim, in spite of the approach, is to satisfy all members and minimise their disagreement towards the recommendation. Some approaches that follow this aim are the *least misery* [162] and *average without misery* [145] approaches, which achieve certain degree of fairness but do not guarantee a high level of agreement among members towards the recommendation.

Given that there are few available public datasets for research in the group recommender systems domain [228], the groups used to evaluate group recommendations are formed in different ways to simulate the aforementioned group notions:

- *Random groups*: Random group formation matches the situation of a number of users who group in order to do an activity [228].
- *Similar groups*: Users group following the principle of homophily, this is, the groups are composed of users with similar features, such as interests, beliefs, education or age.
- *Dissimilar groups*: Users group following the principle of heterophily, this is, the groups are composed of users with diversity of features.

As aforementioned, the aggregation approaches for group recommendation of-

ten overlook the relationships among members' preferences, such as experience overlap or preference similarity. Therefore, in this research a group recommendation model that considers these features of the group in a opinion dynamics model is developed. This way, group specific features are considered when making recommendations.

A known limitation of GRSs is that the recommendations might not meet all members' preferences. In these cases some recommended items are not satisfactory for one or more members of the group. In order to minimise the chance of this situation, the least misery or multiplicative aggregations are applied [162, 146]. This research aims to apply consensus reaching processes to avoid such situations considering all members' preferences.

2.4 Context-aware recommender systems

The quality of recommendation, the usage of item's features, user features and ratings from users over the items might not be sufficient to deliver relevant recommendations in certain domains. For example, in the domain of restaurant recommendation, availability of choices might be reduced regarding the context in which the recommendation is done, e.g., recommending a restaurant for having dinner in an hour or so. The recommender system might find the perfect match for the users that are going to have dinner together, but the restaurant is closed or too far away. To overcome this limitation, context-aware recommender systems [6] take into account contextual features of both the users and the items in order to improve recommendation utility.

Context-aware recommender systems (CARS) are an extension to traditional recommender systems to consider that users' preferences over the items are not influenced only by their own taste, but they are also influenced by the context in which the items are consumed by the user [6]. The consideration of user context leads to an improvement of the utility of the recommendations for the user in certain domains [21], which positively influences the perceived utility of the system

[93].

In CARS, the ratings are provided under certain context, therefore a rating is a tuple ($user$, $item$, $context_1$, ..., $context_n$) $\rightarrow$ $rating$. In order to manage this information, different approaches have been used. Adomavicius et al. [6] consider three approaches (see Figure 2.6)

- *Contextual pre-filtering*, the recommender system uses only the information generated on the target context in order to compute the recommendation. With the contextualized information, a traditional recommender system selects the relevant items. Therefore, the feedback used as input is different regarding the target user's context.

- *Contextual post-filtering*, a traditional recommender system computes the recommendation for the target user regardless the context. A posterior step of refinement adjusts the item predictions in order to consider the features of the target context.

- *Contextual modeling*, in this approach, contextual information is used directly in the recommendation process. This way, instead of using a traditional 2D RS (Item $\times$ User $\rightarrow$ Rating) the CARS is a truly multidimensional function (User $\times$ Item $\times$ Context $\rightarrow$ Rating) [149, 253]

Researchers have compared contextual pre-filtering with contextual post-filtering and found that none of them completely dominates the other ones [6, 172, 173]. Panniello et al. [172] suggest to evaluate first the contextual pre-filtering approach and compare its results against a traditional recommender system. If the pre-filtering method improves the results of the traditional one, then the pre-filtering approach should be used. Otherwise, the efforts should focus on finding the best post-filtering. The rationale for this suggestion is that the contextual pre-filtering approach is easily tested, while in contextual post-filtering there are many alternatives to combine the contextual attributes to adjust the recommendations and finding a suitable one might be difficult.

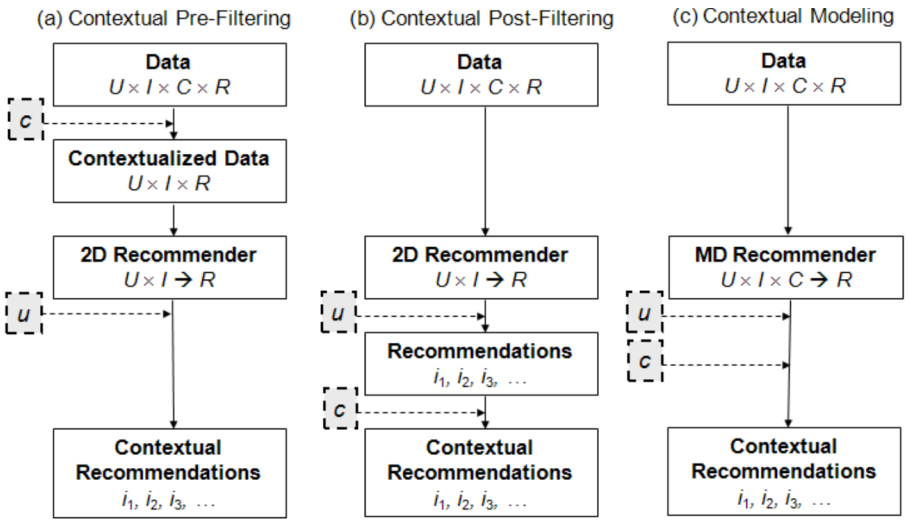


Figure 2.6: Different approaches to integrate the contextual information in the recommendation [6].

The importance of using the context in the recommendation is greater when it comes to recommendation on mobile devices [191]. Mobile devices are capable of providing ubiquitous access to the information, along with the possibility of extracting various contextual features from the sensors that they have built-in. This scenario allows to use the RSs not only for planing [2, 85], but also to request the recommendation in the moment where it is needed.

These facts make mobile CARS suitable to improve the experience of tourists. Tourism activities are strongly influenced by context. For example, a user is doing a route to visit many places in a certain region and he has a schedule to follow. For today, he had planned to visit a castle and then travel to a close town to have lunch at a fancy restaurant. However, there is an unexpected event that makes it necessary to change the plan. In this case, a mobile recommender system that takes into account the context helps the user to reschedule the trip recommending multimedia content, context-aware services, views/ratings of peer users [88].

In RSs for tourism, various contextual dimensions have been pointed out as relevant [88]. Among them, the users' and items' location is important, which can modify the recommendation or even exclude items because they are too far to

reach. Time is also an important contextual dimension, given that the recommending tourist activities should be different if the travel is planned for summer or for winter. Other contextual information are the user's mood, local time, weather, or companion, among others.

2.4.1 Contextual modeling approaches

In contextual modeling, the context is not taken as a variable that influences the traditional information, but as another source of information that is used by the recommender system and influences the recommendation model and, therefore, the recommendations. The most successful approaches for contextual modeling are based on matrix factorization. These techniques build a model with a high number of parameters and try to learn the best values of the parameters using the contextual data.

Sparse Linear methods

The Sparse Linear Method (SLIM) is a predictor based on the nearest neighbors algorithm. SLIM is based on computing the pairwise similarity among items [203] or users [190]. The model of this predictor is the following:

$$\hat{r}_{u_j i_k} = R_{u_j *} \cdot W_{* i_k} = \sum_{l=1, j \neq k}^{N} R_{u_j i_h} W_{i_k i_l} \qquad , diag(W) = 0 \tag{2.8}$$

where W is the matrix with the pairwise similarity among items, which is usually computed with the cosine coefficient of the ratings over both items.

To extend traditional RS, the contextual SLIM (CSLIM) approach considers that there are interactions between users and contexts or between items and contexts [254]. In the latter, the model is the following:

$$\hat{r}_{u_j i_k c} = r_{u_j i_k} + \sum_{l=1}^{L} D_{i_k c_l} \tag{2.9}$$

Therefore, the task of the CSLIM is to find the parameters of D, which can be done using gradient descent. The GCSLIM method considers a general scenario to

integrate both CSLIM-U and CSLIM-I, together with an analysis of the similarity among different contexts [254]. GCSLIM is based on estimating contextual ratings from rating deviations across contexts, and thus learning these deviations in the model training phase.

Tensor factorization

A way to include context-awareness in the traditional matrix factorization is to consider that the items have certain interaction with the context [22]. This way, the loss function is formulated in the following way:

$$\hat{r}_{uic_1\ldots c_k} = \overrightarrow{v}_u \cdot \overrightarrow{q}_i + a_i + b_u + \sum_{j=1}^{k} B_{fc_j} \tag{2.10}$$

where $\overrightarrow{v}_u$ and $\overrightarrow{q}_i$ are the d dimensional vectors associated to the user and item, a_i is the mean rating of the item, b_u is the baseline parameter of the user (user bias), and B_{fc_j} is the parameter modeling the interaction of the contextual conditions and the category of the item. This way, the model has a parameter for each item category and context combination.

Other approaches consider to build more complex models, which results in better prediction power. However, because of the number of parameters to optimise, these models need a huge amount of information in order to be trained properly, which makes them not suitable for problems where the data is scarce. An example of these approaches is the Multiverse Recommendation tensor factorization [118], which allows to consider contextual and non-contextual ratings and to refine the model size regarding the number of users, number of items or number of contexts, i.e., m, n and c. The model is formulated in the following way:

$$F_{u_j i_k c_l} = S \times_U U_{u_j *} \times_I I_{i_k *} \times_C C_{c_l *} \tag{2.11}$$

where $S \in \mathbb{R}^{d_U \times d_I \times d_C}$ is a central tensor, and $U \in \mathbb{R}^{m \times d_U}$, $I \in \mathbb{R}^{n \times d_I}$, $C \in \mathbb{R}^{c \times d_C}$ are matrices of factors for the users, items and context. Their dimension can be different in order to manage the scalability of the model.

2.4.2 Frameworks for context-aware recommender systems

In the literature there are other interesting efforts to facilitate the integration of CARS in a given system providing the implementation of the algorithms [254] or hiding the technical details of the models [150].

In the first case, there are open source projects that provide the implementation of the CSLIM algorithms [254]. Specifically, CARSKit[1] provides the implementation of the algorithms and also includes the evaluation of the algorithms, to allow experimentation in the specific domain.

In the latter scenario, the researchers try to find a way to generalize the context-aware problem to apply their model to any domain of recommendation [150]. Their model allows to use a single software that hides the technical details, which reduces the requirements of statistical and data mining knowledge that the system administrator needs to develop and tune a CARS.

2.4.3 Context-awareness in group recommender systems

A materialization of the separation between CARS and GRS is the approach proposed by Hussein et al. [112, 113]. They presented *Hybreed*, a software tool that allows to develop recommender systems in a fast way focused on the recommendation to individuals taking into account the contextual features [112]. A posterior update of the software incorporated the support of both group and context-aware recommendations [113] through context views, which are a generalization of the context that leads to the usage of virtual contexts. The recommendations are computed within certain virtual context, which is selected taking into account the contextual features. A change in the context of recommendation triggers a change in the virtual context and the recommendations are recalculated to focus on the new context.

It is worth to mention that, also in the integration of CARSs and GRSs, there are systems that focus on the recommendation of groups to users. In this direc-

[1] https://github.com/irecsys/CARSKit

tion, [246] proposes a system for social recommendation with context-awareness, which recommends groups for users to join. The proposal is evaluated in the Flickr dataset, that contains information about groups of users.

The main division that can be extracted from the application of context-awareness in group recommendation is the context source. It can be divided into two categories: (i) environmental context, which uses contextual features extracted from the physical world such as location, weather or time, and (ii) group context, which uses group related context, such as sessions or emotional state.

Environmental context

A number of approaches for group recommendation with external context, and for CARS in general, use the location of the users and items in order to improve the recommendation utility, given the greater accessibility to location sensors with the usage of mobile devices.

An early example of GRS with CARS is the system proposed by Coutand et al. [60]. In this system, they consider an architecture to help users in mobile environment. They determine that, in addition to the recommendation of adequate products, the system addresses the group-awareness, the support of the group management, allows members communications and ensures sufficient privacy and generates user trust.

Other examples of GRS that use the location are those that recommend points of interest (POIs) to groups of users. In this direction, *OmniSuggest* [119] supports recommendation of (POI) to both individuals and groups taking into account contextual information such as POI category, time, location, speed and traffic. It integrates social network data and contextual recommendation, thus, it proposes an approach to integrate all the contextual information. The approach proposed is evaluated with Foursquare data and shows that the consideration of both group recommendation and context-aware recommendation improves the group satisfaction towards the recommendation of POI.

Another example of location-based services for groups is the recommendation of events. In this direction, Smaaberg et al. [210] test a prototype doing a user study to determine the best approach to recommend music events to groups of users. This GRS takes into account the location and the time frame of the events, to avoid recommending past events. Group's members state how satisfied are with the recommendations provided by different algorithms of recommendation. The authors point out that implicit context relaxation can be useful in this scenario.

In addition to the usage of GPS, the GRS with external context use other information provided by sensors. This way, several approaches take into account sensor management and information integration.

An example of system that takes into account environmental context with the usage of sensors is a system that enables dynamic group formation and recommendation based on context acquired using RFID tags [196]. This system identifies the users that are present through RFID tags to personalize the recommendation, relying in the sensors to manage the group. It also allows the activation-deactivation of context-based services through RFID tags. Additionally, the system uses a multisensor board to detect position and movements of the users. The system integrates these sources to detect higher level scopes through the usage of an ontology.

Another possibility to improve the GRSs is to completely rely on advanced sensor capabilities to form the groups. Cassagranda et al. [52] propose to detect the groups of users sharing the same environment analyzing GPS and acoustic fingerprint of users through their mobile device. This way, the system can group together users that are in similar context to complete the experience. The approach is applied to the enrichment of broadcast radio with personalized and context-aware audio content.

More general approaches apply general models to integrate the external context using contextual modeling. In this direction, Stefanidis et al. [213] use a hierarchical context model to introduce context-awareness and allow context generalization. The approach selects the preferences of the user that are in line with the preferences

of the other members of the group. The preferences of the users include contextual information, and the selected ones are those in the target context, in addition to the group similar ones. The hierarchy is used to augment preferences with with context states that specify the situations under which preferences hold.

Internal context

On the other hand, there are some researchers that focus on the context that is related to the group for which the recommendation is delivered, and therefore they take into account contextual information such as member characterization, browse session or emotional state.

In the first case, the GRS can focus on doing a thorough member characterization. In this direction, Zakrzewska et al. [241] analyze the recommendation of e-learning resources to student for learning. The performance of the technique is validated on the basis of data of students, who are described by cognitive traits such as dominant learning style dimensions. In the same domain, Myszkorowski et al. [156] proposes a fuzzy recommender systems for students that takes into account both the context of usage and the colleagues of the target user.

Other branch of works reject the assumption that traditional RSs do about a monolithic user's profile and try to separate the users' preferences to group them by different context, given that the interaction might occur in different situations that can influence the satisfaction. Loeb et al. [132] propose a system to accomplish this feature providing improvements over traditional systems. In order to support this profile fragmentation and later selection of fragments, it collects contextual user information. In the recommendation step, the relevant profile fragments are selected regarding the target context. A study of the system shows that this separation leads to a greater satisfaction of users towards the recommendations, which suggests that the profile fragmentation captures the influence of both the group and context over individual preferences.

Additionally, profile fragmentation has been used to minimize members dis-

satisfaction towards the recommendation. In this direction, dos Santos et al. [72] study the concept of always welcome recommendations, defined as recommending items only in the particular context where they are useful. With this aim they split the profiles to fit them in the context and study the situations in which a user might not be satisfied with a recommendation to propose a system that avoids such situations.

Other source of context is the browse session, that can use the content being played as marker of the current context of the group. This way, Wang et al. [229] propose a system that delivers TV recommendations to the group, which are computed using individual preferences that is later contextualized using the current video being played, to adjust the recommendation to the specific context. This approach shows significantly better performance than traditional methods in situations of high group dynamics and inactive group's members, under different group sizes.

On a completely different kind of GRS with internal information, there are works that use sensors to detect members attitude. This way, the context of the group might be related to the behaviour of group's members when they are exposed to certain content, which can be monitored through the usage of sensors. Kurdyukova et al. [128] study the exposition of content in a screen located in a public space. This screen has a camera to extract implicit ratings and features of the people that are looking, such as gender, expression, face orientation and conversational features. These features are the context that the system takes into account. The system is evaluated in institutional content recommendation and the study shows that there is almost no relation between the detected social context and user satisfaction. In the same way, [216] propose a system that monitors users' behavior integrates it in a content manager TV that decides the program displayed. The system focuses on resolving conflicts that might arise due to differences in individual preferences.

A step further on context detection is the usage of ontologies to extract and infer

contexts. In this direction, Wang et al. [227] proposes a scenario-aware system that fuses multi-source for the scenario modeling and inference, which helps in systems with changing situations. The inference learns the behavior patterns on different contexts using fuzzy logic to improve the recommendation

2.5 Concepts used in the proposal

This section introduces the notions and concepts that are required to understand the novel proposals developed in this research.

2.5.1 Hesitant Fuzzy sets

Hesitant Fuzzy Sets (HFSs), were introduced by Torra [217] as an extension of Fuzzy Sets [240] in which, given a reference set, there are multiple membership functions, which provides a way of modeling hesitation. In this research, HFSs are applied to consider that group's preferences have certain hesitation, and that each individual preference can be a value for such a group hesitation. This section introduces hesitant fuzzy sets concepts.

Definition 1 *[217]. Let X be a reference set, a Hesitant Fuzzy Set (HFS) on X is a function h that returns a non-empty subset of values in [0,1]:*

$$h : X \to \wp([0,1]) \tag{2.12}$$

Moreover, a HFS can be defined as a set of fuzzy sets.

Definition 2 *[217]. Let $M = \{\mu_1, \ldots, \mu_n\}$ be a set of membership functions. The HFS associated with M, h_M, is defined as*

$$\begin{aligned} h_M &: X \to \wp([0,1]) \\ h_M(x) &= \bigcup_{\mu \in M} \{\mu(x)\} \end{aligned} \tag{2.13}$$

Xia et al. [233] completed the original definition of HFS including the concept of Hesitant Fuzzy Element, which is a particular subset of values in [0,1] for a particular $x \in X$.

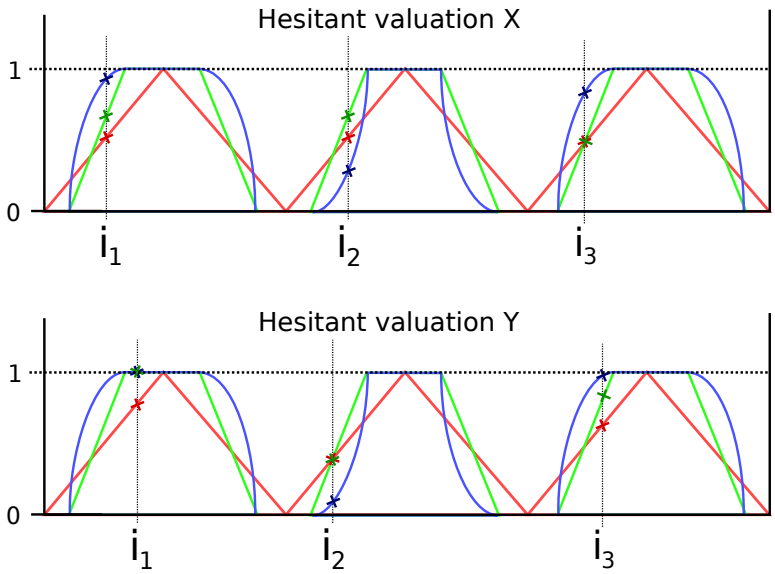


Figure 2.7: Example of hesitant valuations

Definition 3 *[232]. Let X be a reference set, an HFS on X can be represented as*

$$E = \{\langle x, h_E(x)\rangle : x \in X\} \tag{2.14}$$

and the set of values $h_E(x)$, for a particular $x \in X$, is called a Hesitant Fuzzy Element (HFE), which denotes the possible membership degrees of the particular element x.

This way, for each item in X there is a HFE, i.e., a set of membership values in [0,1]. In the application of HFS to group recommendation, the hesitation comes from the cardinality of the group. This is, regarding a given item there is not a unique rating but a set of ratings, one rating for each member. Figure 2.7 shows an example of two valuations, X and Y which belong to two different groups, and three items, i_1, i_2 and i_3. Due to the hesitation that might appear to rate each item, instead of providing only one value, a HFS is used to represent each group valuation.

For applying HFSs in group recommendations, some functions defined for crisp values or fuzzy sets need to be extended. Torra et al. [218] proposed the following extension principle to export operations from fuzzy sets to HFS.

Definition 4 *[218]. Let $E = \{H_1, \ldots, H_n\}$ be a set of n HFS and Θ a function, $\Theta : [0,1]^n \to [0,1]$, we then export Θ on fuzzy sets to HFSs defining*

$$\Theta_E = \bigcup_{\gamma \in H_1(x) \times \cdots \times H_n(x)} \{\Theta(\gamma)\} \tag{2.15}$$

This principle has been applied to the Pearson's correlation coefficient (PCC) [178, 194], a function widely used in RSs. González et al. [92] extended the PCC, noted as ρ, to the hesitant Pearson's correlation coefficient (HPCC), ρ_{HFS}. The correlation between two valuations X and Y is measured by HPCC.

Definition 5 *Let X and Y be two HFSs on S and $h_X(s_i) \times h_Y(s_i)$ be the collection of all pairs of HFEs,*

$\left((h_X(s_i))^{(j)}, (h_Y(s_i))^{(k)}\right)$

where $j \in \{1, \ldots, l_X(s_i)\}$ and $k \in \{1, \ldots, l_Y(s_i)\}$, being $l_X(s_i)$ and $l_Y(s_i)$ the cardinals of $h_X(s_i)$ and $h_Y(s_i)$ respectively.

The set of all pairs HFEs for each $s_i \in S$ is given by,

$$R_{HFS} = \cup_{s_i \in S} h_X(s_i) \times h_Y(s_i) \tag{2.16}$$

where $i \in \{1, \ldots, n\}$.

The number of pairs of values in R_{HFS} is computed as,

$$|R_{HFS}| = \sum_{i=1}^{n} (l_X(s_i) \times l_Y(s_i)) \tag{2.17}$$

Definition 6 *[92]. Let X and Y be two HFSs on S, the hesitant Pearson's correlation coefficient (HPCC), ρ_{HFS}, is defined as follows.*

$$\rho_{HFS}(X,Y) = \frac{SSC(h_X, h_Y)}{\sqrt{SS(h_X)}\sqrt{SS(h_Y)}}, \tag{2.18}$$

where SSC corresponds to the covariance of both sets, which is defined as:

$$\begin{aligned} &SSC(h_X, h_Y) = \\ &= \sum_{i=1}^{n} \sum_{j=1}^{l_X(s_i)} \sum_{k=1}^{l_Y(s_i)} \left((h_X(s_i))^{(j)} - \overline{h_X}\right) \left((h_Y(s_i))^{(k)} - \overline{h_Y}\right), \end{aligned} \tag{2.19}$$

where $\overline{h_X}$ and $\overline{h_Y}$ are the arithmetic mean of the corresponding values of the first and second elements of the pairs, respectively.

$$\overline{h_X} = \frac{1}{|R_{HFS}|} \sum_{i=1}^{n} l_Y(s_i) \left(\sum_{j=1}^{l_X(s_i)} (h_X(s_i))^{(j)} \right) \tag{2.20}$$

$$\overline{h_Y} = \frac{1}{|R_{HFS}|} \sum_{i=1}^{n} l_X(s_i) \left(\sum_{j=1}^{l_Y(s_i)} (h_Y(s_i))^{(j)} \right) \tag{2.21}$$

where $SS(h_X)$ and $SS(h_Y)$ are the standard deviations of the respective sets, which is defined as:

$$SS(h_X) = \sum_{i=1}^{n} l_Y(s_i) \left(\sum_{j=1}^{l_X(s_i)} \left((h_X(s_i))^{(j)} - \overline{h_X} \right)^2 \right) \tag{2.22}$$

$$SS(h_Y) = \sum_{i=1}^{n} l_X(s_i) \left(\sum_{j=1}^{l_Y(s_i)} \left((h_Y(s_i))^{(j)} - \overline{h_Y} \right)^2 \right) \tag{2.23}$$

The hesitant Pearson's correlation coefficient is used in the proposed hesitant group recommender model (HGRM) to perform the group's preference modeling and improve group recommendations quality.

2.5.2 Group decision making and consensus reaching processes

In Group Decision Making (GDM) [135] problems, a set of experts ($E = \{e_1, \ldots, e_p\}$) aims to select a solution from a set of alternatives ($A = \{a_1, \ldots, a_q\}$). This problem happens in diverse contexts, such as certainty, risk, and uncertainty. However, most real decisions happen under uncertainty. The fuzzy preference relation is the most used structure to manage preferences in uncertainty context.

A fuzzy preference relation [175] P_i given by an expert e_i is defined by a membership function $\mu_{P_i} : A \times A \rightarrow [0,1]$. This function can be represented as a square matrix where each μ_i^{kl} denotes $\mu_{P_i}(a_k, a_l)$, which is the preference degree of expert e_i of the alternative a_k over a_l. If this preference degree is greater than 0.5, then it indicates the degree to which a_k is preferred over a_l. If it is lower than 0.5, then it indicates that a_l is preferred over a_k. A $\mu_{P_i}(a_k, a_l) = 0.5$ indicates that according to expert e_i both alternatives are equally preferred.

A selection process is done to retrieve a solution from the set of alternatives af-

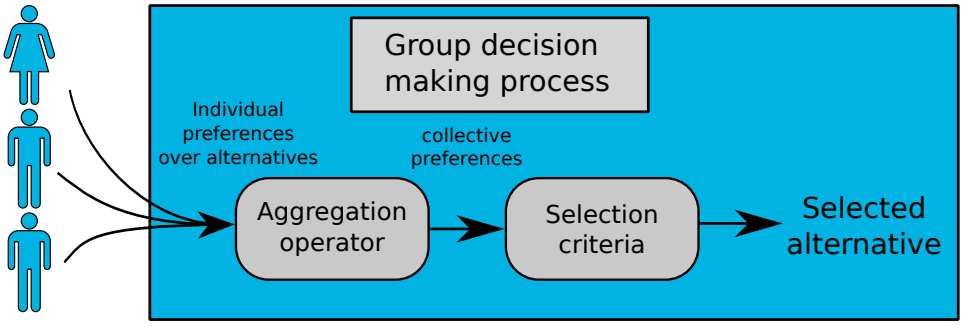


Figure 2.8: Group decision making.

ter all experts expressed their individual preferences as shown in Figure 2.8. However, the sole selection of the best alternative does not guarantee an agreement. This situation might result in then experts not feeling that their opinion has been considered in the process.

Consensus reaching processes [235] were introduced in group decision making to avoid the previous problem and select agreed solutions [200]. Therefore, the aim of a consensus reaching process is to assist experts at reaching certain agreement before the selection of the final alternative. In this research, two automatic consensus reaching models are considered: (i) the minimum cost consensus model, and (ii) the automatic consensus support system model based on feedback.

Minimum Cost Consensus Model

The minimum cost consensus model, developed by Zhang et al. [247] focuses on minimizing costs associated to the modification of independent experts' opinions to reach consensus. Such minimum cost is obtained solving the following lineal

programming model.

$$\begin{cases} min \sum_{u=1}^{n} c_u |\bar{o_u} - o_u| \\ s.t. \bar{o} = \sum_{u=1}^{n} w_u \bar{o_u} \\ |\bar{o_u} - \bar{o}| \leq \varepsilon, u = 1,2,...,n \\ \sum_{u=1}^{n} w_u |\bar{o_u} - \bar{o}| \leq \gamma \end{cases} \tag{2.24}$$

The parameters of this model are:

- c_u is the cost of modifying the preferences of the expert u.
- o_u is the initial preference of the expert u, before the beginning of consensus process.
- $\bar{o_u}$ is the final preference of the expert u, after consensus reaching.
- $\bar{o}$ is the collective preference of the group of experts.
- ε is the maximum possible distance between collective and individual preferences.
- w_u is the weight of the expert u.
- $\gamma = 1 - \alpha$, where α is the acceptable level of consensus

This model fits well into the GRS scenario, given that it retrieves as one of the output a collective preference of the group of experts, which could be assumed as the group's preference value from the GRS point of view.

Automatic consensus support system model based on feedback

In other consensus models, the consensus reaching process is usually done through an iterative discussion among experts, in which they revise their individual prefer-

ences, that finishes when experts' opinions meet the consensus condition (see Fig. 2.9) [110]:

- Consensus measure: The consensus degree among all individual preferences is computed, where $cr \in [0,1]$ is calculated.
- Consensus control: If $cr \geq \mu \in [0,1]$, where μ is the consensus degree required, then the requirements are met and the process finishes. Some consensus reaching processes also set a maximum number of rounds.
- Consensus progress: If $cr < \mu \in [0,1]$, then the process moderator suggests some updates to the individual preferences in order to increase their consensus degree.

The moderator figure supervises the process and carries out the following tasks:

- Assess the agreement level among experts.
- Find alternatives that are far from consensus.
- Suggest preference changes to experts in order to increase the consensus.

In this research, a group recommendation model is extended with the automatic consensus support system model based on feedback [170]. The aim is to deliver agreed group recommendations to increase members satisfaction towards the recommendation.

In order to apply an automatic consensus reaching process, an essential aspect is the definition of the consensus measure, which quantifies the agreement level within the group from experts' preferences. There are several consensus measures proposed in the literature [168]:

- *Consensus measures based on distances to the collective preference [31, 107]:* The collective opinion P_c is calculated aggregating individual experts' preferences. The consensus degrees can be obtained aggregating the distances between the experts preferences and the collective opinion.

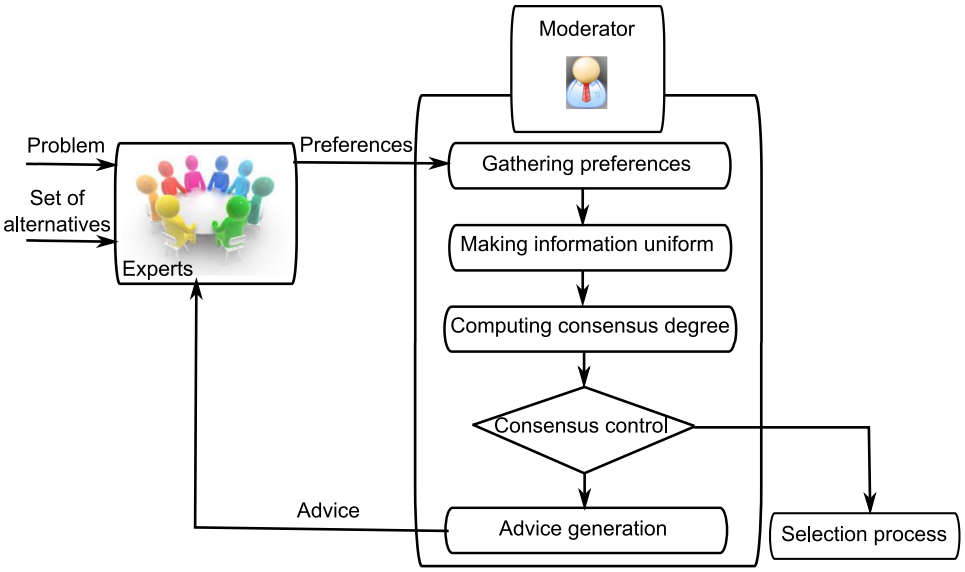


Figure 2.9: Scheme of resolution of a group decision making problem with a consensus reaching process.

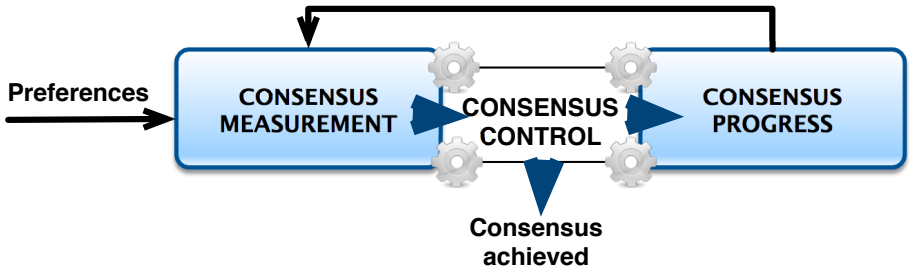


Figure 2.10: General CRP scheme

- *Consensus measures based on distances between experts* [40, 108]: The distance between all pairs of experts is computed and all of them are aggregated to obtain the consensus degree.

Several automatic consensus reaching models have been proposed to provide various features regarding the consensus reaching process [110, 168]. Most of these existing approaches have the following phases:

1. *Consensus Measurement*: The agreement level in the group is calculated from all experts' preferences applying a consensus measure.

2. *Consensus Control*: The consensus degree is compared with the threshold to end the process in the case that the value reaches the threshold, or to continue the consensus reaching process otherwise.

3. *Consensus Progress*: In automatic consensus models, experts preferences are updated automatically. To do so, the model can identify opinions that are far from the colleective preference or identify pairs of experts that are far [31, 231, 247].

The ConsensusGRS proposal developed in this research uses the automatic consensus support system model based on feedback [170] to compute agreed group recommendations.

2.5.3 Opinion dynamics

Opinion dynamics models aim to model specific social behavior aspects within a number of individuals. They model experts group's opinion evolution over time. Several models have been proposed to consider different opinion evolution assumptions:

- *Bounded Confidence* [102]: Experts update their opinion taking into account opinions within a bound [102]. This bound acts as a threshold of the opinions that the member listens to and gets influenced by. Experts' final opinions are conditioned to the opinion bound and the distribution of initial opinions.

- *Deffuant-Weisbuch* [134]: Experts meet in random pairwise encounters, and, after that, they may agree or not [134]. This model is driven by the same idea of bounded confidence, i.e., only experts whose opinion lie within a bound communicate to each other. In each step only two experts are selected randomly and if their opinion is close, they update their opinions.

- *Kinetic exchange* [62]: Some experts may have a special attitude towards changing their opinion, hence their opinion can be either collaborative, strict or contrary.

- *DeGroot* [68]: Experts update their opinion following social influence, hence they assign certain weight to other members' opinion.

Among the opinion dynamics models, this research focuses on the update of members opinions according to social influences [87]. The DeGroot model [68] suits our aim because opinion evolution is driven by a matrix of weights $A = (a_{m_j,m_k})_{(g\times g)}$, where a_{m_j,m_k} is the weight that expert m_j assigns to expert m_k opinion. The weight a_{m_j,m_j} denotes the weight assigned to himself, and therefore is the degree to which members are unwilling to change their initial preference. This matrix is restricted to satisfy that all the weights of an expert must sum to 1, i.e., $\sum_{m_k\in G} a_{m_j,m_k} = 1$. Opinions are updated in steps in the DeGroot model as follows:

$$x_{m_j}^{t+1} = a_{m_j,m_1}x_{m_1}^t + a_{m_j,m_2}x_{m_2}^t + \cdots + a_{m_j,m_g}x_{m_g}^t \tag{2.25}$$

Equation 2.26 shows an example of weighting matrix among five experts m_1 to m_5. This weighting matrix A drives how the members' opinion change over time. Figure 2.11 shows the opinion evolution of these members for the initial opinions $X^0 = \{1.00, 0.75, 0.50, 0.25, 0.00\}$, where members' opinions are modified until stability is reached. As stated by DeGroot [68], the weighting matrix A determines the relationship between the initial and final opinions. The final opinion is a linear combination of the initial opinions of all members in the form $c = \sum \lambda_{m_j} * x_{m_j}^0$. This linear combination coefficients are determined by the normalized left eigenvector of eigenvalue $\lambda = 1$ of the weighting matrix A, which in the example is $\lambda_1 = \{0.00, 0.00, 0.00, 0.50, 0.50\}$. The final opinion is $X^\infty = X^0 * \lambda_1$, which yields $x_{m_i}^\infty = 0.125$ for all $m_i \in G$.

$$A = (a_{m_j.m_k})_{(g\times g)} = \begin{bmatrix} 0.95 & 0 & 0.05 & 0 & 0 \\ 0 & 0.95 & 0 & 0.05 & 0 \\ 0.02 & 0 & 0.95 & 0.03 & 0 \\ 0 & 0 & 0 & 0.98 & 0.02 \\ 0 & 0 & 0 & 0.02 & 0.98 \end{bmatrix} \tag{2.26}$$

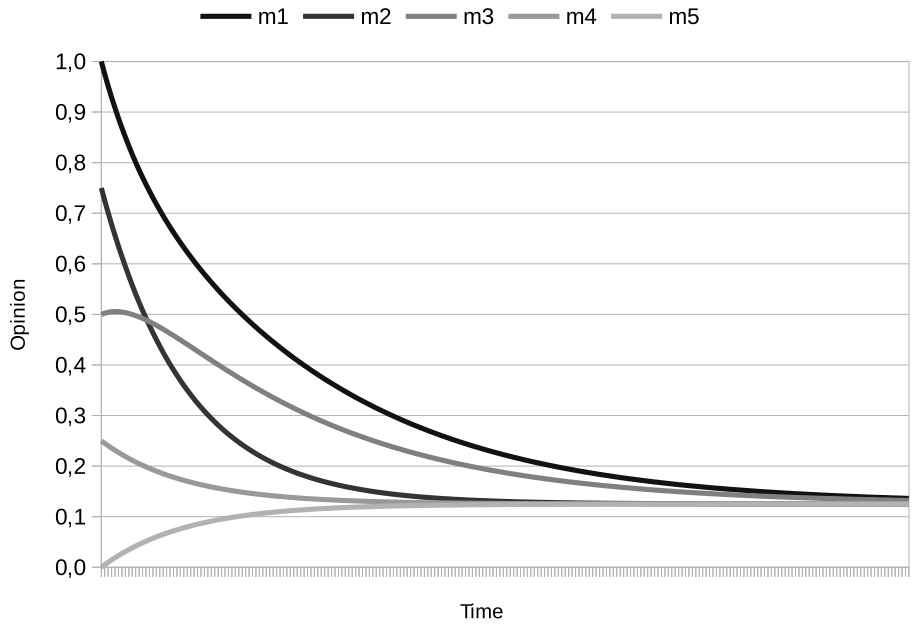


Figure 2.11: Opinion evolution for the weighting matrix shown in Eq. 2.26.

The outcome of the process is as follows. If the left eigenvector is unique, then the final opinion converges to one consensus value as happens in the previous example. If this eigenvector is not unique, then experts' opinion may fragment across several values.

The relationships between members can be expressed in terms of a weighted directed graph $DG(G,S)$ to study opinions evolution in DeGroot's model [70], where $G = \{m_1, \ldots, m_g\}$ is the set of members, and S is the set of directional edges. Thus, s_{m_j,m_k} indicates the degree of the relationship that member m_j has with m_k. The weighting matrix A can be computed from S with Equation 2.27 to apply the DeGroot model.

$$a_{m_j,m_k} = \frac{s_{m_j,m_k}}{\sum_{m_l \in G} s_{m_j,m_l}} \tag{2.27}$$

The normalized eigenvector associated with the left eigenvalue 1 of matrix A is unique as long as there is at least one member accessible by all other members

[70], called the opinion leader. In these cases, all members' opinions converge to a unique value, since all members' final opinions are influenced by the opinion leaders. Given that the aim is to state whether the set of directional edges S leads to consensus, its analysis can be simplified considering the accessibility matrix P (see Equation 2.28). Matrix P can be used to determine whether the weighting matrix leads to consensus [70]. If there is at least one member accessible by all, then the process leads to consensus. Therefore, Equation 2.29 determines if S leads to consensus.

$$P = f(S^{(g-1)}) \quad ,where \quad f(s_{m_j,m_k}) = p_{m_j,m_k} = \begin{cases} 1 & \text{if } s_{m_j,m_k} > 0 \\ 0 & \text{if } s_{m_j,m_k} = 0 \end{cases} \tag{2.28}$$

$$S\ leads\ to\ consensus \iff \exists m_j \ \ s.t. \ \ p_{m_k,m_j} = 1 \ \ \forall m_k \in G \tag{2.29}$$

If S does not fulfill Eq. 2.29, then the opinion fragments across several values. The multiplicity of the left eigenvector, associated with left eigenvalue 1 of the weighting matrix A, determines the number of different opinions at the end of the opinion dynamics process. These eigenvectors are orthogonal, and their normalized form determines how members' opinions are grouped into the disjoint opinion subgroups. Moreover, the normalized eigenvectors determine the contribution of member m_j's opinion to the final value.

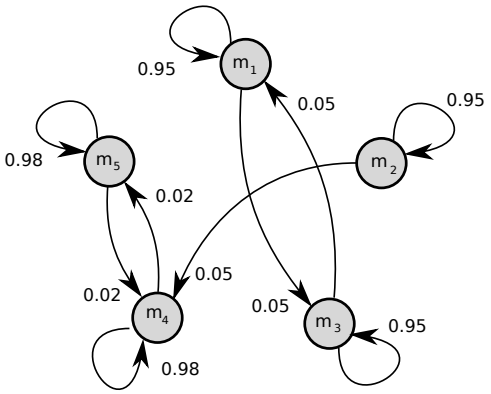


Figure 2.12: Example of similarity matrix of a group expressed as a graph.

Figure 2.12 shows an example of similarities in a group that does not reach consensus. This similarity graph yields the weighting matrix shown in Eq. 2.30, and it has the adjacency matrix shown in Eq. 2.31. As it can be noticed, there is no member accessible by all the remaining members, therefore 2.29 is not fulfilled and the opinion fragments. The opinion evolution for this group is depicted in Figure 2.13. In this example, there are two eigenvectors associated with the eigenvalue $\lambda = 1$, which are $\{0.5, 0.0, 0.5, 0.0, 0.0\}$ and $\{0.0, 0.0, 0.0, 0.5, 0.5\}$ in their normalized form.

$$A = (a_{m_j.m_k})_{(g\times g)} = \begin{bmatrix} 0.95 & 0 & 0.05 & 0 & 0 \\ 0 & 0.95 & 0 & 0.05 & 0 \\ 0.05 & 0 & 0.95 & 0 & 0 \\ 0 & 0 & 0 & 0.98 & 0.02 \\ 0 & 0 & 0 & 0.02 & 0.98 \end{bmatrix} \tag{2.30}$$

$$P = (p_{m_j,m_k})_{(g\times g)} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{2.31}$$

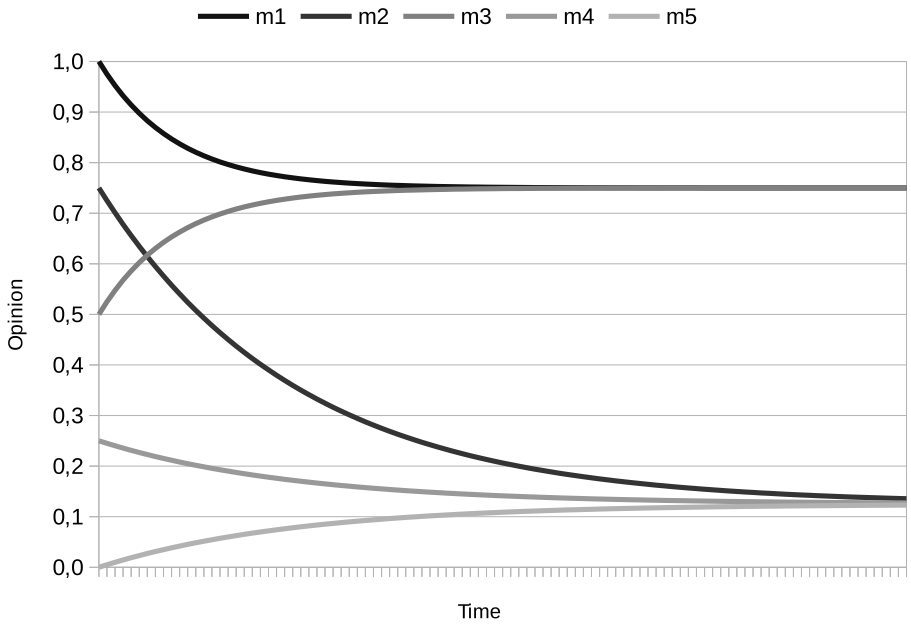


Figure 2.13: Opinion evolution for the weighting matrix shown in Eq. 2.30.

In a weighting matrix A that leads to consensus, the eigenvector is unique. A strictly positive λ_{m_j} in the normalized eigenvector determines the members' opinions that contribute to the final value – the opinion leaders. Therefore, those members whose $\lambda_{m_j} = 0$ simply follow other members opinions. In situations that do not lead to consensus, the eigenvalue has a specific multiplicity, which determines the number of opinion subgroups formed. In the second example, there are two opinion subgroups, $\{m_1, m_3\}$ and $\{m_2, m_4, m_5\}$.

Within each of these opinion subgroups, there is at least one opinion leader [70], which are $\{m_1, m_3\}$ for the first opinion subgroup, and $\{m_4, m_5\}$. Moreover, followers can always reach at least one opinion leader in their respective subgroup, as in the case of m_2. This way, the members belonging to the same subgroup have the same opinion at the end of the process. However, opinions of other subgroups are different, given that they do not influence on each other.

Therefore, a way to ensure consensus is to connect the opinion subgroups adding relationships between leaders of opinion subgroups [70]. This way, their

eigenvectors orthogonality breaks and the connected subgroups have the same opinion at the end of the process. To do this, we can take the leader of any opinion subgroup and create a relationship to any leader of a different opinion subgroup.

By doing so iteratively, all opinion subgroups are connected and the group reaches a consensus value with a modified set of directional edges S'. This process needs to be done at least $q-1$ times, where q is the initial number of opinion subgroups [70]. Different combinations of relationships can be added to ensure consensus.

2.5.4 Natural noise

Several recommendation approaches have been used to improve individual recommendation, such as neighborhood-based collaborative filtering [203], matrix factorization [126], or approaches that consider temporal dynamics [123, 186]. In spite of this, the use of users' preferences has produced some problems that limit their performance, such as *cold-start* and *sparsity* [9, 195], and more recently new related problems regarding the quality of the rating data [130, 179, 239]. A decade ago, it was pointed out that explicitly stated user's preferences may not be error free [164]. More recently, other recent works [30, 55, 97, 248] have also pointed out that a person's ratings might be noisy, inconsistent, and biased. Li et al. [130] determined that too many noisy ratings can distort users' preference profiles, which result in *unlike-minded* neighbors that imply a quality loss in recommendations. Kluver et al. [120] have also suggested that user ratings are imperfect and noisy, and such noise limits the predictive power of any RS. Specifically, Ekstrand et al. [75] pointed out that the rating elicitation process is not error-free, hence the ratings can contain noise. They mentioned that such a noise, previously coined natural noise by O'Mahony et al. [164], could be caused by human error, mixing of factors in the rating process, uncertainty and other factors. Herlocker et al. [103] pointed out that recommendation approaches were reaching *magic barrier* in their accuracy, this is, a lower bound on recommender systems performance due to in-

consistencies and noise in the ratings. Other works [15, 30, 199] stated that their detection and correction should provide more accurate recommendations.

Therefore, in addition to improving recommendations through new recommendation approaches, researchers should also focus on improving the quality of the rating database [16]. In RSs, there are two kinds of noise in the database [164]: (i) *malicious noise*, that consists of erroneous data deliberately inserted in the system to influence recommendations, and (ii) *natural noise*, that appears when users unpurposely introduce erroneous data due to human errors or external factors during the rating process. This research focuses on the latter. Therefore, several approaches have been introduced for managing these rating inconsistencies in recommendation scenarios depending on the information available, such as user dependent approaches [179], item-attributes dependent approaches [13], and also approaches that manage natural noise only using the rating values [239].

The ratings can be gathered implicitly [26] or explicitly, this research focuses on the latter case. Prior research has explored how noisy preferences intentionally inserted by users affect RSs [48, 96], so-called *malicious noise*. However, the noisy ratings introduced unintentionally by users, so-called *natural noise*, has recently attracted the attention of researchers. Several proposals investigate the detection and correction of such natural noise. Some proposals exploit the items' attributes [179], take advantage of user's interaction [16], or use knowledge extracted from the ratings themselves [239]. The main benefit of these proposals is their positive impact on the recommendations.

Previous research presents limitations, such as the removal of information from the dataset [164], or the need of additional information [16, 179]. Specially interesting for this research is the approach proposed by Yera et al. [239] which consists of a two-step method that requires only the rating matrix (see Fig. 2.14):

1. Noise detection: Ratings are tagged as *not noisy* or *possibly noisy* regarding their corresponding user and item behavior (see Fig. 2.15). Each rating and its corresponding user and item are classified as *high*, *medium* or *low*. If the

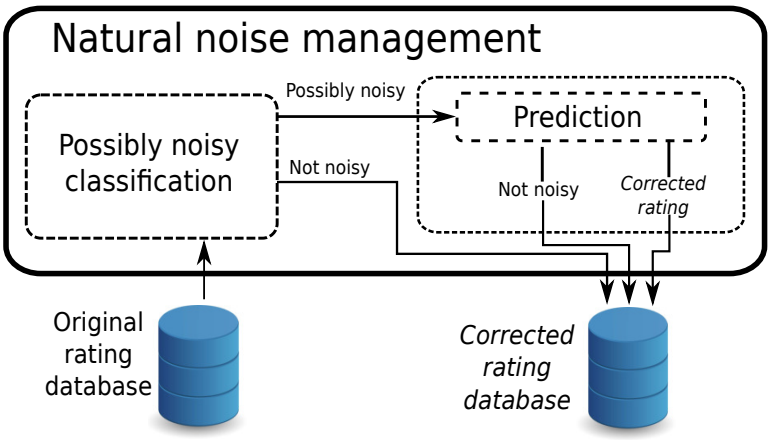


Figure 2.14: General scheme of natural noise management for individuals [239].

Users\Items	low	medium	high	variable
low	PN PN			
medium		PN PN		
high			PN PN	
variable				

PN: Possible noise

Figure 2.15: Classification of the ratings of natural noise management for individuals [239].

user and item behavior are the same and contradict rating classification, then the rating is tagged as possible noise.

2. Noise correction: For each possibly noisy rating a prediction is computed for its corresponding user and item. If the difference between the old and new value exceeds a threshold, then a predicted value replaces the original one.

The major benefit of this approach is that it only needs the information in the rating matrix [239]. A simple illustration of its performance uses the ratings dataset shown Table 2.4. The user and item classes, c_{u_j} and c_{i_k} respectively, are the classes

of the majority of their corresponding ratings. These rating classes are defined as low=1,2 med=3 and high=4,5. If the majority is not absolute, then the class assigned is variable and the rating is not analyzed for noise. In the example, considering users' behavior c_{u_j} and item tendency $R_{\bullet i_k}$, the ratings classified as possibly noisy are r_{u_1,i_3}, r_{u_3,i_3}, and r_{u_6,i_1} because they contradict the user's behavior and item's tendency.

Table 2.4: Illustrative example for the classification of the ratings as possibly noisy.

	U						c_{i_k}
	u_1	u_2	u_3	u_4	u_5	u_6	$R_{\bullet i_k}$
i_1	5	5	5	5	5	2	high
i_2	5	3	5	3	3	3	medium
i_3	3	5	3	5	5	5	high
i_4	5	5	5	5	5	5	high
i_5	1	1	4	2	1	5	low
c_{u_j}	high	high	high	high	high	high	

Natural noise biases recommendations, therefore, its management is a key factor to improve them. There are several Natural Noise Management (NNM) approaches for individual RSs databases. While some NNM approaches need additional information [14, 179], others detect and correct the natural noise using information already contained in the database [237, 239].

So far, natural noise has been studied only in RSs for individuals. However, Group Recommender Systems (GRSs) [67] play an important role in many social activities that require recommendations to be delivered to a group of users, such as watching TV with family, sightseeing with others, or going to the cinema with friends. GRS approaches extend individual RS for recommending to groups aggregating user individual information [144]. Therefore, GRSs use explicit ratings, and natural noise is also present biasing the group recommendation. Consequently, its management might play an important role in the quality of the group recommendations. This research is devoted to the natural noise management (NNM) in GRS to study its influence in group recommendation.

Chapter 3

Hesitant Fuzzy Sets-based Group Recommender System

3.1 Introduction

Traditional GRSs are often built through the extension of individual recommendation models in order to work with groups of users [9]. This extension is usually done aggregating the information of each individual of the group in order to obtain a collective preference or recommendation. However, the aggregation process is not exempt of information loss. This issue is particularly important when the distribution, shape and diversity of individual data is considered. Particularly, the aggregation process produces a loss in the diversity of ratings that diminishes recommendation diversity. Consequently, the performance of GRSs could be improved maintaining in the group recommendation process the maximum amount of information provided by the group's members and push the aggregation process forward to the last recommendation steps.

In order to deliver such a model, Hesitant Fuzzy Sets (HFS) concept is applied. HFSs are an extension of Fuzzy Sets [240] introduced by Torra [217]. In HFSs, multiple membership function can be defined, which yield several membership values, instead of a single one. This multiplicity can be used to model hesitation. Therefore, we can consider that group's preferences have certain hesitation, and that each individual preference can be a value for such a group hesitation.

RSs are often evaluated through *accuracy* metrics [9]. Researchers have recently highlighted the importance of considering other recommendation aspects, such as *diversity* [221]. The diversity of a recommendation indicates how dissimilar are the items contained in the recommendation. It is clear that the consideration of diversity would lead to less accurate recommendations, as demonstrated by Zhou et al. [255], therefore the accuracy and diversity of recommendations must be considered together in order to deliver better recommendations. Our aim is to apply HFSs to push forward the information aggregation and avoid information loss in initial recommendation steps, therefore, recommendations diversity is considered to evaluate the proposal.

To achieve such an aim, we propose Hesitant Group Recommender Model (HGRM), which is based on collaborative filtering and hesitant fuzzy sets. HGRM recommends to groups of users avoiding the aggregation process in the first recommendation steps, which maintains the information throughout the group recommendation process and delays the aggregation step in order to provide both accurate and diverse recommendations.

The remainder of this chapter is structured as follows. The Hesitant Group Recommender Model (HGRM) is described in Section 3.2. The experiments and results are demonstrated in Section 3.3. A summary is provided in Section 3.4.

3.2 Hesitant Group Recommender Model

When the target of a CF RS is not a single user but a group of them, the strategy for computing similarities between individual users for finding the neighborhood must be adapted to avoid the aggregation of members' preferences as the first step. In this proposal, the nearest neighbors of a group of users are found, therefore, the way of finding neighbors is adapted to compare individual users with the group profile. To do so, HFS are applied to model group's preferences. Consequently, our proposal builds on top of the user-based collaborative filtering approach and replaces the Pearson's correlation coefficient by the HPCC (see Equation 3.1), which

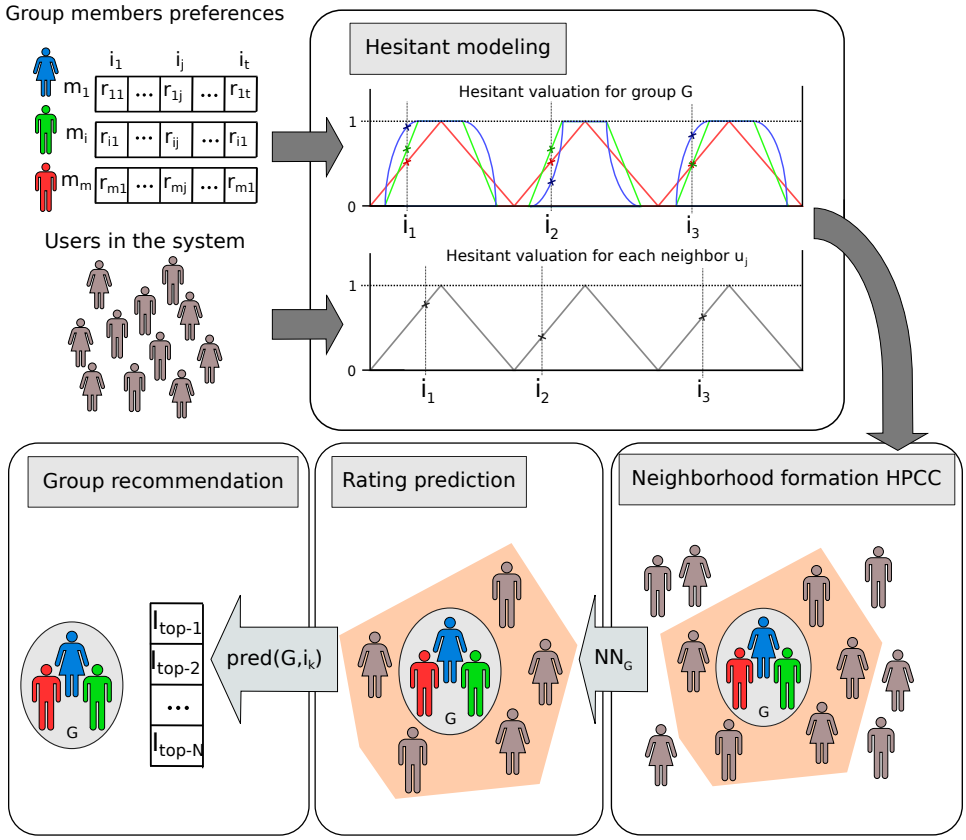


Figure 3.1: General scheme of HGRM

supports the comparison of individual's profiles with group's profiles.

Figure 3.1 depicts the general scheme of HGRM. In general, the proposal follows a similar scheme to the user-based collaborative filtering, with the difference of a target group's neighborhood being computed using all group's preferences without a previous aggregation step. The HGRM proposal scheme consists of four phases:

1. *Hesitant modeling of group's and user's preferences.* The preferences of group's members and other users are expressed as hesitant fuzzy sets.

2. *Neighborhood formation with HPCC.* The nearest neighbors algorithm is modified to provide the set of the K nearest neighbors to the group G using

HPCC.

3. *Rating prediction*. Group G neighborhood NN_G is used to predict ratings for unseen items using neighbors' ratings.

4. *Group recommendation*. The top-n items with highest rating prediction are recommended to the group.

The remainder of this section details each HGRM phase.

3.2.1 Hesitant Modeling of group's and user's preferences

The first phase of HGRM consists of modeling group's and user's preferences with Hesitant Fuzzy Sets in order to avoid aggregating members' preferences as the first step, which would imply a loss of information. For such a endeavor, group's and user's profiles are defined in terms of HFSs:

- A group's profile defined in HFS, X_G, allows to deal with multiple ratings provided over one item by the members of the group G. Hence, X_G is a HFS that contains the ratings given by group G members:

$$X_G = \{\langle i_k, h_{X_G}(i_k)\rangle : i_k \in I\}$$
$$h_{X_G} : I \to \wp([0,1])$$
$$h_{X_G}(i_k) = \left\{\widehat{r_{u_g i_k}} \quad s.t. \quad u_g \in G\right\}$$

  where $\widehat{r_{u_g i_k}}$ denotes the normalized rating $r_{u_g i_k}$.

- A user's profile defined in HFS, Y_{u_j}, expresses user u_j ratings in HFS:

$$Y_{u_j} = \left\{\langle i_k, h_{Y_{u_j}}(i_k)\rangle : i_k \in I\right\}$$
$$h_{Y_{u_j}} : I \to \wp([0,1])$$
$$h_{Y_{u_j}}(i_k) = \left\{\widehat{r_{u_j i_k}}\right\}$$

  where $\widehat{r_{u_j i_k}}$ denotes the normalized rating $r_{u_j i_k}$.

3.2.2 Neighborhood formation with HPCC

The second phase of HGRM consists of building the neighourhood of the group using HPCC. HPCC was proposed by Gonzalez et al. [92], and it extends the Pearson's correlation coefficient [178, 194] to work over Hesitant Fuzzy Sets. Here, we use the HPCC to compute the correlation between the target group's preferences and any other user's preferences.

Once we have defined as HFSs the group's preferences, h_G, and user's preferences, $h_{Y_{u_j}}$, the HPCC [92] between them is defined as follows:

$$\rho_{HFS}\left(X_G, Y_{u_j}\right) = \frac{SSC\left(h_{X_G}, h_{Y_{u_j}}\right)}{SS(h_{X_G}) * SS(h_{Y_{u_j}})}, \tag{3.1}$$

where SSC corresponds to the covariance of both HFSs and is defined as:

$$\begin{aligned} &SSC\left(h_{X_G}, h_{Y_{u_j}}\right) = \\ &= \sum_{i_k}^{I} \sum_{u_g}^{G} \sum_{u_j}^{\{u_j\}} \left(\left(h_{X_G}(i_k)\right)^{(u_g)} - \overline{h_{X_G}} \right) \left(\left(h_{Y_{u_j}}(i_k)\right)^{(u_j)} - \overline{h_{Y_{u_j}}} \right), \end{aligned} \tag{3.2}$$

and $SS(h_X)$ and $SS(h_Y)$ denote the standard deviation of the corresponding sets defined as:

$$SS(h_{X_G}) = \sqrt{\frac{1}{|h_{X_G}|} \sum_{i_k}^{I} \sum_{u_g}^{G} (h_{X_G}(i_k)^{(u_g)} - \overline{h_{X_G}})^2} \tag{3.3}$$

$$SS(h_{Y_{u_k}}) = \sqrt{\frac{1}{|h_{Y_{u_k}}|} \sum_{i_k}^{I} \sum_{u_j}^{\{u_j\}} (h_{Y_{u_j}}(i_k)^{(u_j)} - \overline{h_{Y_{u_k}}})^2} \tag{3.4}$$

where $\overline{h_{X_G}}$ and $\overline{h_{Y_{u_j}}}$ note the average of the HFE values of each set, respectively.

$$\overline{h_{X_G}} = \frac{1}{|h_{X_G}|} \sum_{i_k}^{I} \sum_{u_g}^{G} (h_{X_G}(i_k)^{(u_g)}) \tag{3.5}$$

$$\overline{h_{Y_{u_k}}} = \frac{1}{|h_{Y_{u_k}}|} \sum_{i_k}^{I} \sum_{u_j}^{\{u_j\}} (h_{Y_{u_j}}(i_k)^{(u_j)}) \tag{3.6}$$

With this similarity defined over HFSs, we can build the neighborhood of group G computing the similarity between X_G and each other user's profile Y_{u_k}. The K users with the highest similarity to the group, noted as NN_G, compose the neighborhood of group G. Massa et al. [142] proved that negative correlations do not lead to good results, therefore, neighbors with negative similarity are not included in NN_G.

3.2.3 Rating prediction

In the rating prediction phase, NN_G preferences are used to calculate the rating prediction for group G for each item. In the neighborhood formation phase (see Section 3.2.2), the aggregation of group's preferences is avoided to compute a neighborhood with all the group information. In this phase, it is not necessary to avoid the aggregation of NN_G preferences because NN_G is built considering all information of the group. Therefore, NN_G preferences can be aggregated without a major impact in recommendation diversity.

There are various strategies proposed originally for individual collaborative filtering to compute the predicted rating using the neighborhood [204].

- *Direct prediction*: The ratings of NN_G over the target item are aggregated using a weighted average that considers their similarity to the target group.

$$pred\,(G,i_j) = \frac{\sum_{u_k \in NN_G} \rho_{HFS}\,(X_G, Y_{u_k}) \cdot r_{u_k i_j}}{\sum_{u_k \in NN_G} \rho_{HFS}\,(X_G, Y_{u_k})} \tag{3.7}$$

- *Compensated prediction*: Users may have different bias when rating, such as optimistic users or critical ones. In order to compensate these differences, the user bias is subtracted from the rating before the weighted aggregation. This precition strategy is used in all techniques compared in the experiment.

$$pred\,(G,i_j) = \overline{r}_G + \frac{\sum_{u_k \in NN_G} \rho_{HFS}\,(X_G, Y_{u_k}) \cdot \left(r_{u_k i_j} - \overline{r}_{u_k}\right)}{\sum_{u_k \in NN_G} \rho_{HFS}\,(X_G, Y_{u_k})} \tag{3.8}$$

where $\overline{r}_G$ is the average value of the set of ratings of group's members.

3.2.4 Group recommendation

Once a prediction has been computed for each item, the system outputs a sorted list of items regarding their rating prediction. The recommendation is composed of the $top-N$ items with the highest rating prediction.

3.3 Experimentation and evaluation

This section describes the experiment performed to evaluate HGRM and compare it with aggregation-based GRS models. First, the techniques compared are detailed. After that, the dataset is described. Later, the evaluation measures are defined. Eventually, experiment results are shown and analyzed.

3.3.1 Techniques compared

In order to determine the suitability of HGRM, it is compared with the traditional rating aggregation-based GRS. Here we consider two versions of it: (a) Mean-based pseudo-user GRS, and (b) RMSMean-based pseudo-user GRS. In addition to the comparison to traditional techniques, we also evaluate two versions of HGRM: *HGRM* considers the duplicate preferences as such, and *HGRM no-dup.* eliminates the duplicate preferences. In summary, four models are evaluated in the experiment:

- *Mean*: Pseudo-user GRS with Mean as preference aggregation.
- *RMSMean*: Pseudo-user GRS with RMSMean as preference aggregation.
- *HGRM*: HGRM that considers duplicate preferences.
- *HGRM no-dup.*: HGRM that eliminates duplicate preferences.

The output of each of these techniques is a sorted list of recommended items. In the experiments, we considered top-5 recommendations. The dataset is split in training and test set performing a 20 executions 5-cross fold validation.

3.3.2 Data set

The techniques are compared for the dataset ml-100k [1], which consists of 1682 items, 943 users and 100k ratings. In the dataset, users evaluate movies in the *five stars* domain. In order to work with HFSs, the rating domain is normalized.

The MovieLens dataset contains only individual preferences, but there is no information about the groups. This exprimentation focuses on random groups, which is the most challenging type of groups for GRSs. With random group formation we intend to model the situation of a number of users who group together in order to do an activity [228]. The techniques are compared regarding various group sizes ranging from 1 to 500 users. For the sake of clarity, only the results for groups of size 20, 25, 50, 100, 200 and 500 are shown, which is a representative subset of the group sizes considered.

3.3.3 Evaluation measures

With the proposal of HGRM we aim to maintain information of the group in the recommendation process. In order to evaluate the impact of such aim in the recommendation results, we consider various viewpoints regarding recommendation quality: accuracy, rank quality, and diversity. Three evaluation measures aim to quantify recommendation quality regarding these points of view [50, 89]: *Normalized Root Mean Squared Error*, *Normalized Discounted Cumulative Gain* and *Intra List Similarity*, respectively. These measures are defined as follows:

- Normalized Root Mean Squared Error (NRMSE) [61] measures the deviation in the predictions from the rating true value, where the error is normalized to [0,1], hence, the lower its value, the more accurate is the RS.

$$NRMSE = \sqrt{\frac{1}{N}\sum_{r_{ui}\in R_{test}}\left(\frac{\tilde{r}_{ui}-r_{ui}}{d_{max}-d_{min}}\right)^2} \tag{3.9}$$

- Normalized discounted cumulative gain (NDCG) [23] measures how close

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

is the ranking outputted by the RS to the perfect possible ranking regarding the true rating value. For such a comparison, the utilities of both lists are compared. Its value ranges from 0 to 1, and NDCG=1 is the perfect ranking.

$$DCG = \sum_{k=1}^{N} \frac{\tilde{r}_{ui} - 1}{log_2(k+1)}$$
$$NDCG = \frac{DCG}{IDCG} \tag{3.10}$$

where *IDCG* is the *DCG* of the items sorted by its true rating. Notice that in the results we refer to $1 - NDCG$, therefore, all measures are minimized.

- Intra List Similarity (ILS) [256] measures how similar are the items in the recommendation. Diversity is a desired feature, therefore, the less *ILS* the better.

$$ILS\left(\tilde{I}\right) = \frac{\sum_{i_j \in \tilde{I}} \sum_{i_k \in \tilde{I}, j \neq k} cosine\left(v_j, v_k\right)}{2} \tag{3.11}$$

where v_j and v_k are, respectively, feature vectors of items i_j and i_k, which are computed applying SVD [126] with 20 features over the rating matrix.

The results of the techniques are analysed first independently for each of the three measures. After that, in order to balance the aspects that each measure considers in the selection of the best technique, we combine the *NRMSE* and the ILS, therefore, we have a measure that combines prediction accuracy and diversity. This is particularly important because, as stated by Zhou et al. [255], accuracy and diversity cannot always be improved at once and the improvement over one measure negatively impacts the other.

3.3.4 Experiment results

In this section, the results regarding each evaluation measure are shown and analyzed separately. After that, the combination of NDCG and ILS is also shown and analyzed to balance accuracy and diversity.

Table 3.1 shows results for NMRSE. The results are shown for group sizes ranging from 20 to 500 and the best results are highlighted in bold. As it can be

noticed, the larger the group size, the greater the prediction error. If we compare the results among techniques, both configurations of the HGRM approach show a light decay in rating prediction accuracy.

Table 3.1: Results for NRMSE of Mean, RMSMean, HGRM, and HGRM no-dup across various group sizes.

	Size 20	Size 25	Size 50	Size 100	Size 200	Size 500
Mean	**0.25558**	**0.25572**	**0.25620**	**0.25665**	**0.25747**	**0.25890**
RMSMean	0.25561	0.25576	0.25624	0.25669	0.25751	0.25894
HGRM	0.25591	0.25609	0.25659	0.25702	0.25781	0.25926
HGRM no-dup.	0.25603	0.25620	0.25676	0.25728	0.25815	0.25996

Table 3.2 shows results for 1-NDCG. The results are shown for group sizes ranging from 20 to 500 and the best results are highlighted in bold. These measures should be minimized, i.e., the lower value, the better. As it can be noticed, the larger the group size, the lower ranking quality. However, the results show that all techniques compared report similar values of NDCG to the 4th decimal position, hence, there is no significant ranking quality decay among techniques compared.

Table 3.2: Results for $1-NDCG$ of Mean, RMSMean, HGRM, and HGRM no-dup across various group sizes for a recommendation list of 5 items.

	Size 20	Size 25	Size 50	Size 100	Size 200	Size 500
Mean	**0.07250**	**0.07252**	**0.07256**	**0.07247**	**0.07265**	**0.07266**
RMSMean	**0.07250**	**0.07252**	**0.07256**	**0.07247**	**0.07265**	**0.07266**
HGRM	**0.07250**	**0.07252**	**0.07256**	**0.07247**	0.07266	**0.07266**
HGRM no-dup.	**0.07250**	**0.07252**	**0.07256**	**0.07247**	**0.07265**	**0.07266**

Table 3.3 shows the results for ILS and the best results are highlighted in bold. In general, all techniques compared show less ILS when the group size increases. It is worth to remark that for HGRM and HGRM no-dup. *the bigger the group size the more ILS decay*, and that their magnitude of decay is greater than for Mean or RMSMean techniques. Therefore, HGRM and HGRM no-dup. deliver more diverse recommendations.

Table 3.3: Results for *ILS* of Mean, RMSMean, HGRM, and HGRM no-dup across various group sizes for a recommendation list of 5 items.

	Size 20	Size 25	Size 50	Size 100	Size 200	Size 500
Mean	**0.84599**	**0.83347**	**0.78712**	0.71462	0.63706	0.44160
RMSMean	0.84833	0.83739	0.79293	0.71737	0.63849	0.44423
HGRM	0.85172	0.84111	0.79121	0.71535	0.62920	**0.38945**
HGRM no-dup.	0.85216	0.84227	0.79437	**0.71180**	**0.62109**	0.40837

As aforementioned, it is needed to check the results across all evaluation measures to properly decide which is the best technique. As shown in Table 3.2, NDCG results do not vary across techniques, therefore, we can ignore the ranking quality in the combined analysis of measures. Thus, accuracy and diversity are analyzed together using a convex combination of NRMSE and ILS, where $\alpha \in [0,1]$ is the importance of accuracy over diversity:

$$NRMSE \cdot \alpha + ILS \cdot (1-\alpha) \tag{3.12}$$

The techniques compared have been evaluated for $\alpha \in \{0.75, 0.50, 0.25\}$. These values correspond to 3/1, equal, and 1/3 importance of accuracy over diversity, respectively. The results for each α are shown in Tables 3.4, 3.5 and 3.6, respectively. In the same way, Figures 3.2a, 3.2b and 3.2c show the results for the three different combinations.

Table 3.4: Results for $NRMSE \cdot \alpha + ILS \cdot (1-\alpha)$, $\alpha = 0.75$

	Size 20	Size 25	Size 50	Size 100	Size 200	Size 500
Mean	**0.40318**	**0.40015**	0.38893	0.37114	0.35236	0.30457
RMSMean	0.40379	0.40116	0.39041	0.37185	0.35275	0.30526
HGRM	0.40486	0.40234	**0.39024**	0.37160	0.35065	**0.29180**
HGRM no-dup.	0.40505	0.40271	0.39116	**0.37090**	**0.34888**	0.29706

The results on the combined study show that HGRM and HGRM no-dup. obtain better values when the group size increases as compared to the results of Mean and RMSMean models, i.e., traditional aggregation-based models. This tendency

Table 3.5: Results for $NRMSE \cdot \alpha + ILS \cdot (1-\alpha)$, $\alpha = 0.50$

	Size 20	Size 25	Size 50	Size 100	Size 200	Size 500
Mean	**0.55078**	**0.54459**	**0.52166**	0.48563	0.44726	0.35025
RMSMean	0.55197	0.54657	0.52458	0.48702	0.44799	0.35158
HGRM	0.55381	0.54859	0.52390	0.48618	0.44350	**0.32435**
HGRM no-dup.	0.55409	0.54923	0.52556	**0.48453**	**0.43961**	0.33416

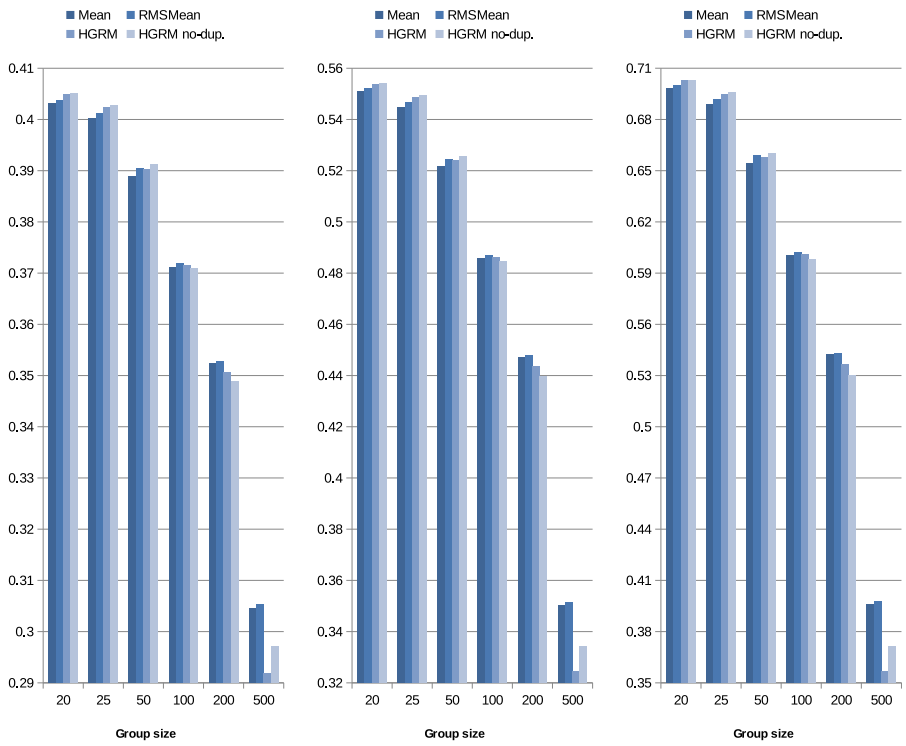
Table 3.6: Results for $NRMSE \cdot \alpha + ILS \cdot (1-\alpha)$, $\alpha = 0.25$

	Size 20	Size 25	Size 50	Size 100	Size 200	Size 500
Mean	**0.69839**	**0.68902**	**0.65439**	0.60012	0.54216	0.39592
RMSMean	0.70015	0.69198	0.65875	0.60219	0.54324	0.39790
HGRM	0.70276	0.69485	0.65755	0.60076	0.53635	**0.35690**
HGRM no-dup.	0.70312	0.69575	0.65996	**0.59816**	**0.53035**	0.37126

is clear from groups of 100 members an more, and it suggests that avoiding the initial aggregation step in user-based neighborhood approaches for group recommendation improves recommendation diversity while maintaining good accuracy results.

As it can be observed for all α values used, HGRM and HGRM no-dup. achieve a remarkable difference as compared to the traditional models as the group size increases, showing clear improvements for group size greater than 100 users. This difference suggests that the improvements of HGRM and HGRM no-dup. in terms of recommendation diversity do not have a negative influence in the accuracy of the system. Consequently, HGRM and HGRM no-dup. balance accuracy and diversity for large groups, which makes both approaches suitable for large group recommendation in recommendation domains where diversity is important.

In conclusion, the accuracy and diversity are properly balanced in HGRM and HGRM no-dup. when recommending to large groups, which makes them suitable in contexts with a relative importance of accuracy over diversity less or equal to 3. These results confirm the hypothesis of keeping all information from group's members avoiding aggregation processes, the GRS performance will improve tak-



(a) $\alpha = 0.75$ (b) $\alpha = 0.50$ (c) $\alpha = 0.25$

Figure 3.2: Results for $NRMSE \cdot \alpha + ILS \cdot (1 - \alpha)$

ing into account different properties.

3.3.5 Computational complexity

The evaluated approaches were analyzed regarding their computational complexity. Table 3.7 details the computational complexity of each task within the compared approaches, and Table 3.8 shows the computational complexity of each approach. The Mean and RMSMean have been grouped as Aggregation-based GRS, and HGRM and HGRM no-dup. have been grouped as HGRM-based GRS. In these tables, $|U|$ is the number of users, $|I|$ is the number of items, $|G|$ is the group size, R_G is the number of ratings of the group G, $|R_G^{max}|$ is the largest amount of ratings of a member within G, and k is the neighborhood size.

As shown in both tables, both methods have the same computational complex-

Table 3.7: Computational complexity of proposal tasks.

Task	Complexity	Aggr. GRS	HGRM
Group aggregation	$\mathcal{O}(\|G\|\|R_G^{max}\|)$	✓	
Hesitant group modeling	$\mathcal{O}(\|R_G\|)$		✓
Neighborhood formation (agg.)	$\mathcal{O}(\|U\|\|I\|)$	✓	
Neighborhood formation (HPCC)	$\mathcal{O}(\|U\|\|I\|\|R_G\|)$		✓
Rating prediction	$\mathcal{O}(k\|I\|)$	✓	✓

Table 3.8: Computational complexity of evaluated approaches tasks.

GRS approach	Computational complexity
Aggregation-based GRS	$\mathcal{O}(max(\ \|G\|\|R_G^{max}\|\ ,\ \|U\|\|I\|\ ,\ k\|I\|))$
HGRM-based GRS	$\mathcal{O}(max(\ \|R_G\|\ ,\ \|U\|\|I\|\|R_G\|\ ,\ k\|I\|))$

ity with minor differences. The Group aggregation task has $\mathcal{O}(|G||R_G^{max}|)$ and Hesitant group modeling has $\mathcal{O}(|R_G|)$. Although the computational complexity order is similar, if we consider that $|R_G| \leq |G||R_G^{max}|$, then the Hesitant group modeling perform less operations. Regarding the neighborhood formation with HPCC, it has a greater computational complexity order because HPCC computes the cartesian product between the ratings of the group and the potential neighbor for each item. Considering all tasks, the neighborhood formation dominates the computational complexity. However, this greater computational complexity order of HGRM-based GRS is justified by the better performance obtained in terms of accuracy and diversity.

3.4 Summary

This chapter outlines a model of GRSs based on HFSs to model the hesitation of group's preferences, with the aim of delaying the information aggregation process to avoid the loss of information associated to it. The performance of the proposal has been evaluated and validated in an experiment that compares various techniques for group recommendation based on collaborative filtering. The results show that the application of Hesitant Fuzzy Sets in recommender systems for managing members' preferences increases recommendation diversity for large groups. In large groups, diversity of recommendation is important because the more diverse

the recommendation, the greater chance of covering different or conflicting interests of the group's members. Therefore, with HGRM there is less chance of having a recommendation with no acceptable item for a member of the group because of the increased recommendation diversity while maintaining good accuracy results.

Chapter 4

Consensus-driven Group Recommender Systems

4.1 Introduction

Previous chapter proposed HGRM, a model for group recommendation that avoids to aggregate members' preferences in the initial phase of the recommendation process, which is proven to be a good step towards delivering better group recommendations. However, HGRM does not consider group dynamics that may influence group recommendation, such as the negotiation of individual preferences when making a group decision. This situation has been previously studied in group decision making with consensus reaching processes, where experts negotiate their preferences in order to select an agreed alternative. The aim of the consensus process is to avoid group's members not satisfied with the recommendation. Previous works have atempted to achieve this feature in group recommendation using the minimum operator for the recommendation aggregation process [162]. However, the application of this operator only guarantees a minimum level of agreement on the recommendation, but it does not guarantees an acceptable level of agreement among members over the group recommendation.

This chapter aims to study consensus reaching processes, from group decision making, from the point of view of group recommendation. With consensus reaching processes we aim to provide group recommendation with the ability to deliver

group recommendations that satisfy group's members as a whole, and also we aim to give GRSs the added value of reaching an acceptable agreement level among the users regarding the group recommendation. Hence, group recommender systems are extended integrating consensus reaching processes for group decision making. In group decision making, a common solution is found by several individuals, or experts, among a set of alternatives, or possible solutions to the problem [49, 135]. To do so, each expert elicits their preferences regading each alternative. The traditional selecction processes for GDM problems [106] overlook that some experts might disagree with the alternative selected. This situation is mitigated applying consensus reaching processes [109, 168], in which a high level of agreement is achieved before selecting the final alternative. In order to bring experts' preferences closer, they iteratively change them to make them closer to each other. Often, a certain level of agreement is required before making group decisions [200].

Therefore, this chapter proposes a group recommendation framework to provide agreed recommendations applying automatic consensus reaching models. Specifically, the minimum cost consensus model and the automatic consensus support system model based on feedback are considered. The minimum cost consensus model computes the agreed group recommendation modifying the individual preferences constrained by a cost function, which is solved applying linear programming. The automatic consensus support system model based on feedback simulates the interaction between group's members and a moderator that suggests changes to individual recommendations to bring them closer and reach a high consensus level on the recommendations before producing the group recommendation. Both models are evaluated and compared with baseline techniques in the experimentation. For the automatic consensus support system model based on feedback, a graphical visualization, which uses MENTOR tool [171] based on self-organising maps [121], of the consensus between the recommended items and the collective preference is shown.

The remainder of this chapter is structured as follows. The the consensus-

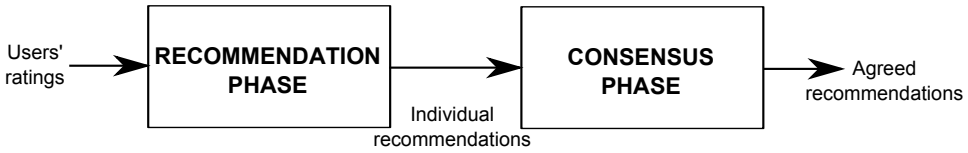


Figure 4.1: General scheme of the consensus-driven group recommender system

driven group recommender system for agreed recommendations is described in Section 4.2. The experiments, its results and a case study are demonstrated in Section 4.3. A summary is provided in Section 4.4.

4.2 Consensus-driven Group Recommender System

In this section, we introduce a novel consensus-based framework for group recommender systems that uses individual recommendations to deliver group recommendations under a high level of consensus. The general scheme of the framework is depicted in Figure 4.1, and it is composed of the following phases:

1. *Recommendation phase*: Individual recommendations are computed for each member using a collaborative filtering method. Later, these individual recommendations are filtered to find the $top-N$ common sets of items, which are represented as preference orderings to be used in the following phase.

2. *Consensus phase*: The individual recommendations are then used in an automatic CRP until a predefined level of consensus is reached. Finally, the collective preference is calculated and the group recommendation is computed.

4.2.1 Recommendation Phase

In this phase, the individual recommendations are computed first for each member. The output for each $m_j \in G$ is a collection of pairs $i_k, \tilde{r}_{m_j i_k}$ for each $i_k \in I - I_{m_j}$. Such recommended items are sorted by rating prediction in descending order. This sorting establishes an ordering O_{m_j} for each member m_j. Given that users have

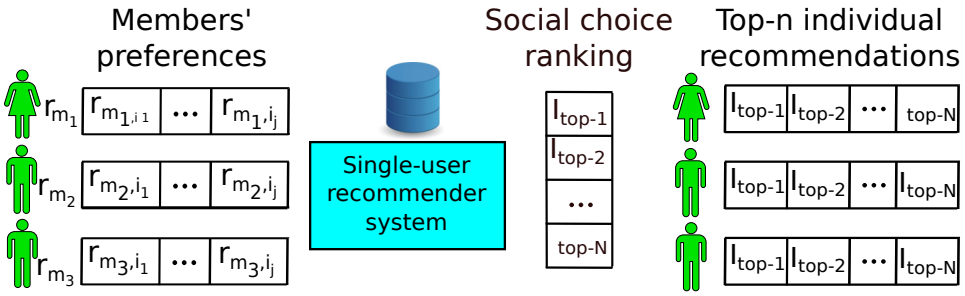


Figure 4.2: Detail of the recommendation phase of the consensus-driven group recommender system.

rated different items, their orderings will contain different item sets. In order to fix this, only the commonly recommended items are considered in the ordering O_{m_j}, this is, only items in set $I - \cup_{m_j}^{G} I_{m_j}$ are considered. Lastly, the top-n items are selected to proceed to the following phase. The steps that compose this phase are depicted in Figure 4.2, and are further detailed in the remaining of this section.

The input of the system is the set of all ratings in the recommender system. It is important to notice that, although Figure 4.2 depicts a complete rating matrix, in the usual recommendation scenario only a small subset of items from the possible ones are known, this is, $R \subset I \times U$.

In the first step, single user collaborative filtering is applied. First, all individual predictions are generated for each $m_j \in G$ using their ratings over the items in the GRS database. These individual predictions are generated for items not previously rated by the member, this is, all predictions $\tilde{r}_{m_j i_k}$ are generated for $i_k \in I - I_{m_k}$.

After this, it is needed to extract the common set of recommended items. This process is done because the single user recommender system may not be able to generate predictions for all $<m_j, i_k>$ pairs, which is not acceptable for the later consensus phase. The set of commonly predicted items for the group G is referred as $I_{\tilde{R}_{G\bullet}}$, and it is defined as:

$$I_{\tilde{R}_{G\bullet}} = \{i_k \quad s.t. \quad \forall m_j \in G \quad \exists \tilde{r}_{m_j i_k}\} \tag{4.1}$$

Once the set of comonly predicted items has been computed, the predictions are

pre-filtered in order to reduce the item set that the consensus phase will use. This is done to shorten the set of recommended items and reduce the computational cost of the consensus phase. As a side effect, this selection discards items that are bad candidates for the group recommendation. For the selection, a number of social choice voting systems are applicable, such as Borda count [39], cumulative voting [19] or single transferrable voting [74].

4.2.2 Consensus Phase

In previous phase, the individual recommendations of each member regarding the common set of recommended items are generated. The consensus phase aims to obtain the collective preference with a high level of consensus using the individual recommendations. Among the available automatic consensus support models, this chapter explores the application of two: (i) the minimum cost consensus model, and (ii) the automatic consensus support system model based on feedback. Both models are detailed in the remainder of this section.

Minimum cost consensus model

This variant of the consensus phase applies the minimum cost consensus model [247] to the individual predictions obtained in the recommendation phase. For each member, this phase receives their individual predictions, which are considered as individual preference values in this model. The minimum cost consensus model adjusts users' preferences such that consensus is reached. Once consensus is reached, the model retrieves one preference value for each item, which is the group rating prediction used to recommend.

A separate instance of the minimum cost consensus model is applied to each item i_k. In each instance, the following rules drive the minimum cost consensus model:

- Member m_j preference is the rating prediction for the target item $\tilde{r}_{m_j i_k}$.
- The cost of modifying member m_j preferences is $c_{m_j} = 1$.

- Member m_j weight is $w_{m_j} = 1/|G|$.
- $\varepsilon = 0.2$, $\alpha = 0.8$.

Therefore, the minimum cost consensus model [247] solved with linear programing is the following:

$$\begin{cases} min \sum_{m_j \in G} c_{m_j} |\hat{r}_{m_j i_k} - \tilde{r}_{m_j i_k}| \\ |\hat{r}_{m_j i_k} - \hat{r}_{G i_k}| \leq \varepsilon, \forall m_j \in G \\ s.t. \quad \hat{r}_{G i_k} = \sum_{m_j \in G} w_{m_j} \hat{r}_{m_j i_k} \\ \sum_{m_j \in G} w_{m_j} |\hat{r}_{m_j i_k} - \hat{r}_{G i_k}| \leq (1 - \alpha) \end{cases} \tag{4.2}$$

where $\hat{r}_{m_j i_k}$ is member m_j preference over item i_k at the end of the consensus process, and $\hat{r}_{G i_k}$ is the collective preference over item i_k at the end of the process.

After this model is solved, the collective preference is used as the target item i_k prediction for the group:

$$Prediction(G, i_k) = \hat{r}_{G i_k} \tag{4.3}$$

It is worth to mention that the consensus model can be applied only to the $top-k$ items from the social choice ranking. In this case, the average value is used for the group prediction for items that are not in the $top-k$ set.

Automatic consensus support system model based on feedback

In this variant, an automatic consensus support system simulates the interaction between group's members and a moderator that suggests updates to bring their preferences closer to each other [170]. After a certain level of consensus is reached, the group's preference is used to compute the group recommendation. The scheme of the consensus phase is depicted in Figure 4.3, and it is detailed in the remaining of this section.

The first step is to express individual predictions as fuzzy preference relationships to be used by the consensus model. Therefore, the individual predictions are

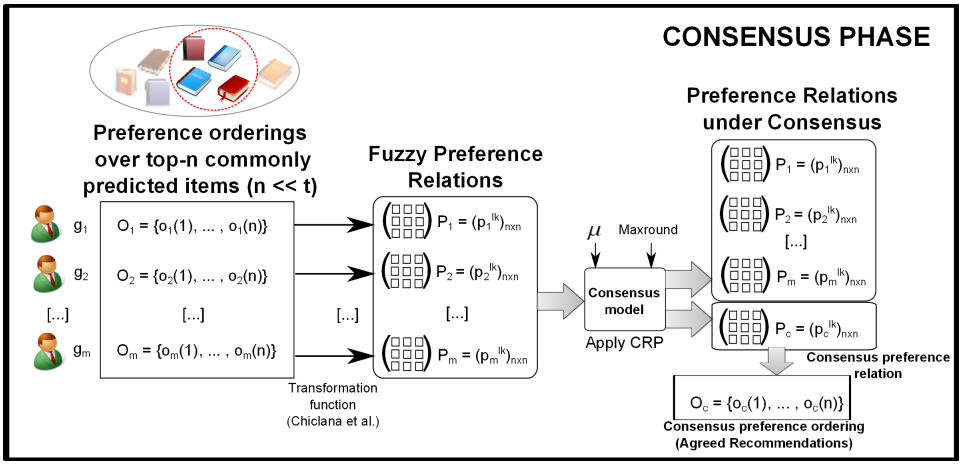


Figure 4.3: Consensus phase with the automatic consensus support system model based on feedback.

converted in crisp orderings. Each ordering $\tilde{O}_{m_j}$ is expressed as a fuzzy preference relation [170] P_{m_j} using a transformation function. Chiclana et al. [58] proposed several transformation functions to deal with various preference representations in decision making problems under uncertainty. Specifically, they developed a transformation function to build the fuzzy preference ordering from a crisp preference ordering:

$$P_{m_j} = (p_{m_j}^{i_k i_l})_{(n \times n)} \tag{4.4}$$

$$p_{m_k}^{i_k i_l} = \frac{1}{2}\left(1 + \frac{\tilde{o}_{m_j}(i_l) - \tilde{o}_{m_j}(i_k)}{n-1}\right) \tag{4.5}$$

The first step is exemplified as follows. $\tilde{O}_{m_1} = \{i_3, i_2, i_4, i_1\}$ is the recommendation for member m_1. This recommendation is expressed in a preference ordering where $\tilde{o}_{m_1}(i_3) = 1$ notes that item i_4 has the highest ranking for member m_1 because it had the highest prediction. Its corresponding fuzzy preference relation is

the following:

$$P_{m_1} = \begin{pmatrix} - & 0.33 & 0.67 & 0.17 \\ 0.67 & - & 0.83 & 0.33 \\ 0.33 & 0.17 & - & 0 \\ 0.83 & 0.67 & 1 & - \end{pmatrix}$$

where $p_{m_1}^{i_1 i_2}$ is computed, according to Eq. (4.5), as:

$$p_{m_1}^{i_1 i_2} = \frac{1}{2}\left(1 + \frac{\tilde{o}_{m_1}(i_2) - \tilde{o}_{m_1}(1)}{4-1}\right) = \frac{1}{2}\left(1 + \frac{2-3}{3}\right) = 0.33 \tag{4.6}$$

The second step of the consensus phase is to use the fuzzy preference relation P_{m_j} of all members and conduct a CRP in order to bring these preferences closer to each other progressively until the consensus level reaches the required value. Here, an automatic consensus model is applied, which automatically updates the preferences to bring them closer to each other using a feedback mechanism that suggests preference updates to individuals [168], following the scheme depicted in Figure 2.10.

The CRP begins measuring the consensus in the group regarding their fuzzy preference relations. A similarity matrix $SM_{m_j m_k} = (sm_{m_j m_k}^{i_l, i_m})_{n \times n}$ is computed for each pair of group's members, where $sm_{m_j m_k}^{i_l, i_m}$ is the similarity between members m_j and m_k regarding their assessment for items i_l and i_m. Once the similarity matrices are obtained, the CRP computes the consensus matrix $CM = (cm^{m_j m_k})_{n \times n}$ through pairwise aggregation. Each pairwise aggregation is done applying an aggregation operator, such as the arithmetic mean or the OWA operator [169], to the similarity values. Finally, when the consensus matrix CM is computed, the overall consensus degree $cr \in [0,1]$ is obtained aggregating the values of CM.

The CRP continues checking the consensus level in the group to determine whether group's members have reached enough agreement. This is done comparing the cr and the $\mu \in [0,1]$, which is a value fixed a priori that determines the minimum degree of agreement required. If the consensus degree cr is equal to or greater than μ, then there is enough agreement among member preferences and

the process finishes. Otherwise, the CRP proceeds to update members' preferences. Furthermore, the number of rounds of update are limited by the parameter *Maxrounds*.

The core method of CRP is to perform the consensus progress. This part evaluates all fuzzy preferences matrices to determine group's members whose fuzzy preference matrix P_{m_j} are furthest from consensus. First, a collective preference P_c is obtained aggregating individual assessments on each pair of items. For each $m_j \in G$, a proximity matrix $PP_{m_j} = (pp_{m_j}^{i_k i_l})_{(n\times n)}$ is computed, where $pp_{m_j}^{i_k i_l}$ indicates the closeness of the member opinion to the collective preference regarding each pair of items [169]. After that, members whose preferences are not close enough to consensus are identified using the proximity matrix PP_{m_j}. This process identifies preferences $p_{m_j}^{i_k i_l}$ whose value is far from consensus, which are updated automatically [170] to increase the group consensus level cr.

After the consensus progress, the CRP continues with a new round and performs the consensus measurement. The process finishes when, either the consensus level required has been reached or the maximum number of rounds has been exceeded. The collective preference P_c is used to compute the agreed recommendations. Given that a CRP has been applied to compute P_c. it reflects a high level of agreement among group's members.

Finally, an exploitation phase is performed to select the alternatives regarding the collective preference P_c. For this selection, the non-dominance selection criterion proposed by Orlovsky et al. [166] is used. This criterion computes a non-dominance degree for each item using P_c, which determines the strict preference relation for the group $\tilde{P}_c = (\tilde{p}_c^{i_k i_l})_{n\times n}$.

$$\tilde{p}_c^{i_j i_k} = \begin{cases} p_c^{i_j i_k} - p_c^{i_j i_k} & \text{if } p_c^{i_j i_k} > p_c^{i_k i_j}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.7}$$

After this, the non-dominance degree $ND(i_k)$ is computed for each i_k as:

$$ND(i_k) = 1 - \max_{i_k}\{\tilde{p}_c^{i_k i_j}\} \tag{4.8}$$

The non-dominance degree $ND(i_k)$ of each item i_k determines its final ranking in the group recommendation.

4.3 Experimentation and evaluation

This section describes the experiment performed to evaluate both variants of the proposed famework and compare them with previous approaches, such as the average or the least misery [162], to test whether the consensus reaching process improves group recommendation.

The remainder of this section is structured as follows. First, the techniques compared are described. After that, the datasets and methods for processing them are detailed. Later, the evaluation measures are defined. Eventually, the results for both experiments are shown and analyzed. Finally, an example is shown to visually demonstrate the group agreement effect for the group recommender system with the automatic consensus support system model based on feedback.

4.3.1 Techniques compared

The aim of the experiments is to compare the consensus framework for group recommendation with other techniques focused on delivering group recommendations that are satisfactory for all members.

In the first experiment, the minimum cost consensus model variant is evaluated for two single user recommender systems: the user-based collaborative filtering (UBCF), and the item-based collaborative filtering (IBCF). This model is evaluated with 3 configurations and compared with two other techniques, which results in five techniques compared:

- *MinCost top-10:* The minimum cost consensus model is applied only to the 10 best items according to the social choice voting system to obtain the final

aggregate rating value. For the remaining items, the average value is used.

- *MinCost top-50*: Applies the minimum cost consensus model to the top 50 items according to the social choice voting system.

- *MinCost All*: The minimum cost consensus model is applied to the entire set of commonly predicted items, therefore, it disregards the results of the social choice voting system.

- *Mean*: The collective rating is computed with the average of individual predictions for the target item.

- *Minimum*: The collective rating is the minimum of individual predictions for the target item.

In the second experiment, the automatic consensus support system model based on feedback variant is evaluated and compared with the recommendation aggregation GRS with the minimum as the aggregation operator [162]. The individual recommender system used is the UBCF. This RS has been improved with several tweaks [41]. In this experiment, the Pearson's correlation coefficient is used as similarity measure. Given that data sparsity can bias the similarity, a relevance factor is applied to penalize similarities not computed with a sufficient number of co-rated items. The specific relevance factor value used is 20. In the rating prediction step, the weighted sum is used to aggregate neighbors' ratings. For the sake of results comparability, the same selection of top-n items carried out in the consensus-driven GRS is applied. In order to isolate the impact of the CRP itself, the proposal uses exactly the same individual RS. Several configurations for the CRP are evaluated with various consensus degree thresholds μ. In the experiment, we show the results for consensus degree values of $\mu = \{0.8, 0.85, 0.9\}$.

4.3.2 Data set

In these experiments, the Movielens dataset is used. It was collected by the GroupLens Research Project[1] at the University of Minnesota. Specifically, we use the ml-100k version, and it consists of a hundred thousand ratings statements given by 943 users over 1682 movies in $\{1,2,3,4,5\}$ domain.

As it happens in the experiment described in previous chapter, the MovieLens dataset does not contain group information. Therefore, the group formation technique applied is the random group formation, in which the group size is set to five members.

The dataset is split in training and tests sets using the hold-out validation with a 20% test set. Multiple executions of this partition have been performed to obtain reliable results. The hold-out technique has been adjusted for group recommendation selecting the ratings in the test set only from the items rated by each group.

4.3.3 Evaluation measures

Three widely used evaluation measures are applied in these experiments to evaluate the results of the framework regarding its ability to recommend: (i) area under receiver operator characteristic curve, (ii) precision, and (iii) MAE.

Area under receiver operating characteristic curve (AUC) is used to evaluate classifiers results regarding a threshold. In recommender systems, the threshold considered is the recommendation list size. Specifically, the AUC measures how the sensitivity and specificity behave when the threshold increases. This increase generates several points defined by its specificity and sensitivity. These points define a curve whose area is the AUC of the classifier. The greater its value, the better the results of the GRS.

Precision [103] is used to determine how accurate are the recommendations delivered by the RS. Specifically, it measures the ratio of true positive items in the recommendation. Similarly to AUC, the greater its value, the better the results of

[1] http://grouplens.org/

	MinCost top-10	MinCost top-50	MinCost all	Mean	Minimum
UBCF	0.6424	**0.6445**	0.6433	0.6428	0.6251
IBCF	**0.5553**	0.5429	0.5437	0.5527	0.5494

Table 4.1: Evaluation results according to AUC. Larger values indicate better performance.

	MinCost top-10	MinCost top-50	MinCost all	Mean	Minimum
UBCF	0.7744	0.7746	0.7748	**0.7742**	0.8684
IKNN	**0.7992**	0.8142	0.8147	0.7995	0.9171

Table 4.2: Evaluation results according to MAE. Smaller values indicate better performance.

the GRS.

Mean Absolute Error (MAE) is used to determine the precision of the rating prediction of a group recommender system. It measures error, therefore, the lower its value the better the group recommender system.

4.3.4 Experiment 1: group recommender system with the minimum cost consensus model

Tables 4.1 and 4.2 show the performance achieved by the compared techniques regarding AUC and MAE, respectively. The best results are highlighted in bold. In the case of AUC (see Table 4.1), the best performance was achieved by MinCost top-50 with UBCF and for IBCF the best performance was achieved by the MinCost top-10 approach. In the case of MAE (see Table 4.2), the best performance with UBCF was achieved by Average and for IBCF the best performance was achieved by MinCost top-10. It is worth to mention that the Minimum technique achieved much worse results regarding MAE as compared with the remaining techniques. It is also worth to mention that the MinCost top-10 and MinCost top-50 achieves better results as compared to MinCost all, which means that the application of automatic consensus techniques on a reduced set not only reduces

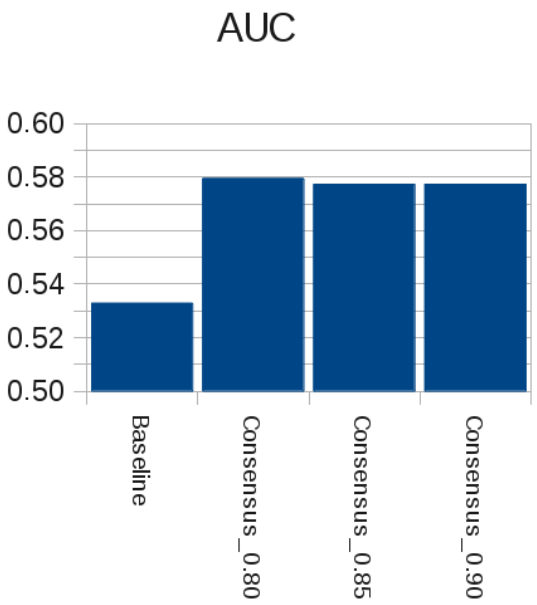


Figure 4.4: Results of area under receiver operating characteristic curve (AUC) for evaluated configurations.

the computational cost, but also improves the performance. Overall, consensus models improve group recommendations in the majority of cases, and for the case of MAE with UBCF, consensus models achieve a similar performance to the best technique.

4.3.5 Experiment 2: group recommender system with the automatic consensus support system model based on feedback

Figure 4.4 shows the results of the compared techniques regarding their AUC. The three configurations of *Consensus* improve the baseline results, which suggests that the CRP benefit the recommendation. Specifically, the best results are obtained for a consensus degree of 0.8 in this dataset.

Figure 4.5 shows the results of the compared techniques regarding their Precision. The recommendation list size is shown in X axis, while the precision of such a list size is shown in Y axis. As it can be noticed, the various techniques compared generate different results regarding precision, which is an evidence of

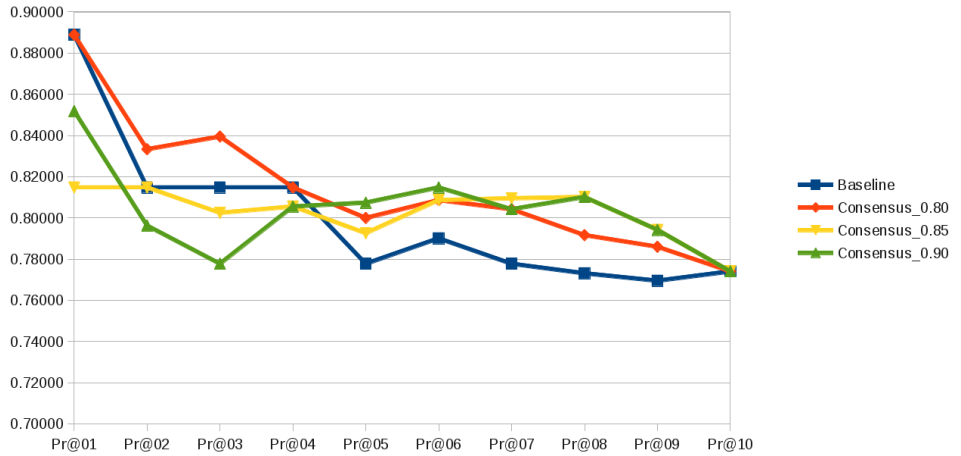


Figure 4.5: Precision at certain recommendation list sizes for evaluated configurations.

each technique is producing different recommendations. Specifically, the proposal with consensus degree of 0.8 shows the best performance for a recommendation list size of up to four items. In the remaining cases, the precision is not the best as compared to the other configurations, but it overcomes the results of the baseline.

It is worth to remark that the results of Precision are calculated for the same set of 10 items. Therefore, the change in the recommendation is the sorting that each technique yields for the ten items. Given that the precision does not consider the ordering of the items recommended, it outputs the same value when the recommendation list size is ten. The precision value at 10 recommendations indicates that there were 77.4% positive ratings in the test set.

4.3.6 Graphical Visualization

The consensus framework for group recommendation changes members' preferences to reach consensus. A graphical visualization of the results is an intuitive manner of showing how members' preferences change and compare individuals positions on both the baseline and the proposal. This section shows an illustrative example of the individuals opinions regarding the collective preference with the automatic consensus support system model based on feedback variant of the

proposed framework.

Previous proposals for recommendation visualization aim to present the information in an intuitive way. Kagie et al. [117] proposes a visualization to highlight similarities between items recommended. This is done making similar items closer in the visualization. In this graphical visualization we have a simlar aim, but instead of depicting only items similarities we aim to higlight where is the collective preference regarding the recommended items. This problem has been already addressed in GDM with self-organizing maps [121]. Specifically, the tool MENTOR is used [171].

Five users are selected from MovieLens dataset, whose data is used to compute recommendations. The individual recommendation lists, similarly to those of the experiment, are composed of ten items. The selected items $\tilde{I}_{\tilde{R}_{G\bullet}}$ are the following:

- i_1: Three Colors: Red.
- i_2: The Fugitive.
- i_3: A space Odyssey.
- i_4: Manon of the Sprius.
- i_5: Delicatessen.
- i_6: Nikita.
- i_7: The princess bride.
- i_8: Raiders of the lost Ark.
- i_9: Return of the Jedi.
- i_{10}: The Godfather II.

After the recommendation phase, an initial ranking of the items is obtained from the individual recommendations. After this phase, a CRP re-ranks the recommendations, which improves agreement. Table 4.3 shows both rankings as

columns. The first column shows the ranking of the baseline group recommendation and the second column shows the ranking of the consensus-based GRS as agreed recommendations. For instance, ”Return of the Jedi” is the first recommendation in the baseline GRS ranking, while it is the second for the consensus-based GRS. It swaps positions with ”The princess bride”. In the case of ”The Fugitive” there is a greater change: in the baseline GRS, it is the fourth and it moves to the ninth position in the consensus-based GRS.

Table 4.3: Group recommendations generated by the baseline GRS and the consensus-based GRS

Baseline recommendation	**Agreed recommendation**
Return of the Jedi	The princess bride
The princess bride	Return of the Jedi
Manon of the Sprius	Manon of the Sprius
The Fugitive	Three Colors: Red
Raiders of the lost Ark	Raiders of the lost Ark
The Godfather II	Nikita
Three Colors: Red	A space Odyssey
Nikita	The Godfather II
Delicatessen	The Fugitive
A space Odyssey	Delicatessen

Figure 4.6 shows the visualization of the recommendations regarding the items recommended. As it can be noticed, the agreed recommendation is located in the center of the graphic, while the baseline recommendation is further to some items. This behavior happens because the consensus-based GRS uses distances in order to make members opinions closer to the collective preferences. In the case of the baseline GRS, it applies the minimum aggregation to avoid the recommendation of less satisfactory items without considering individual preferences as a whole in the aggregation process.

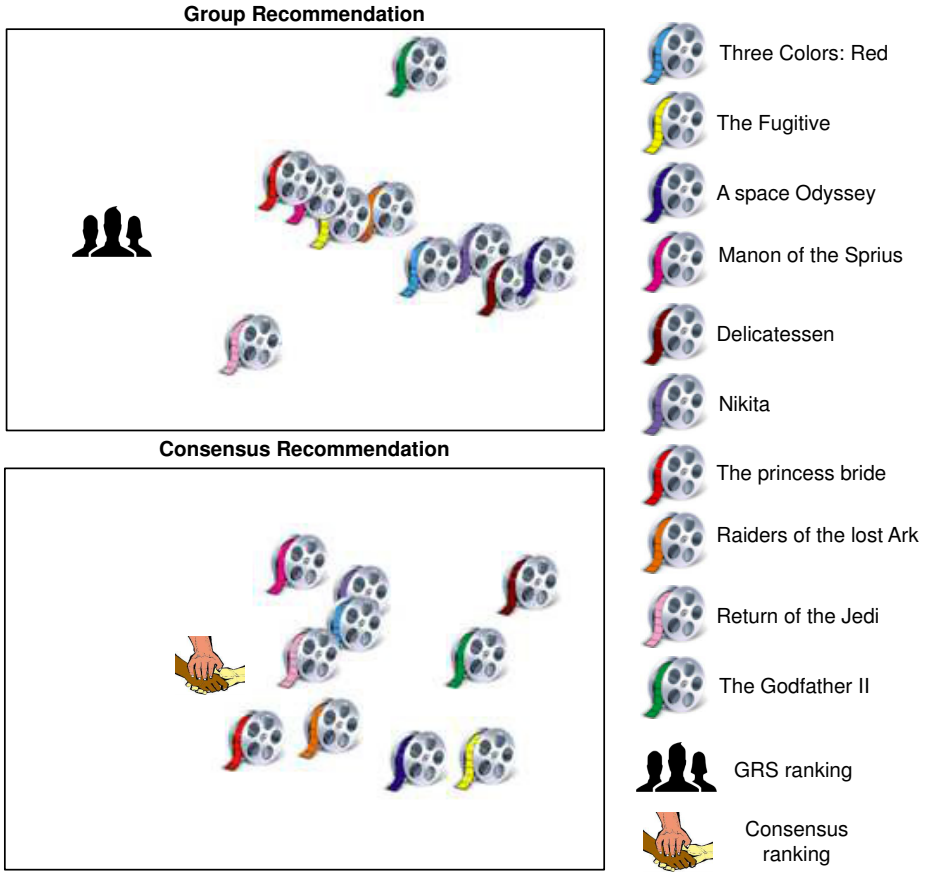


Figure 4.6: Visualization of the recommendations made by the baseline GRS and the consensus-based GRS.

4.3.7 Computational complexity

The evaluated approaches were analyzed regarding their computational complexity. Table 4.4 details the computational complexity of each task within the compared approaches, and Table 4.5 shows the computational complexity of each approach. In these tables, $|U|$ is the number of users, $|I|$ is the number of items, $|G|$ is the group size, R_G is the number of ratings of the group G, $|R_G^{max}|$ is the largest amount of ratings of a member within G, k is the number of rounds taken to reach consensus, and t is the number of items that the consensus model considers.

As shown in both tables, the methods compared have the same computational

Table 4.4: Computational complexity of proposal tasks.

Task	Complexity	UBCF GRS	IBCF GRS	MinCost UB GRS	MinCost IB GRS	ACSSMWF GRS
UB recommendation	$\mathcal{O}(\|G\|\|U\|\|I\|)$	✓		✓		✓
IB recommendation	$\mathcal{O}(\|I\|\|R_u\|)$		✓		✓	
Social choice voting	$\mathcal{O}(\|G\|\|I\|)$	✓	✓	✓	✓	✓
Minimum cost consensus	$\mathcal{O}(\|G\|^2\|I\|k)$			✓	✓	
ACSSMWF	$\mathcal{O}(\|G\|^2t^2k)$					✓

Table 4.5: Computational complexity of evaluated approaches tasks.

GRS approach	Computational complexity
UBCF GRS	$\mathcal{O}(max(\ \|G\|\|U\|\|I\|\ ,\ \|G\|\|I\|\ ,\ \ ))$
IBCF GRS	$\mathcal{O}(max(\ \|I\|\|R_u\|\ ,\ \|G\|\|I\|\ ,\ \ ))$
MinCost UB GRS	$\mathcal{O}(max(\ \|G\|\|U\|\|I\|\ ,\ \|G\|\|I\|\ ,\ \|G\|^2\|I\|k\ ))$
MinCost IB GRS	$\mathcal{O}(max(\ \|I\|\|R_u\|\ ,\ \|G\|\|I\|\ ,\ \|G\|^2\|I\|k\ ))$
ACSSMWF GRS	$\mathcal{O}(max(\ \|G\|\|U\|\|I\|\ ,\ \|G\|\|I\|\ ,\ \|G\|^2t^2k\ ))$

complexity than their individual recommender system with the added task for the Social choice voting. However, UBCF and IBCF do not perform a consensus task. Between the consensus tasks, Minimum cost consensus and ACSSMWF have different computational complexity. However, the greater computational complexity order of consensus-based approaches MinCostUB, MinCostIB and ACSSMWF is justified by the better group recommendation quality.

4.4 Summary

This chapter introduces a framework for consensus-driven group recommender systems, which integrates consensus reaching processes in the recommendation process to provide the added value of improving members' satisfaction towards the recommendation. Two variants of the framework are presented. The first one uses minimum cost consensus model to improve consensus minimizing the changes needed for it. The second one uses the automatic consensus support system model based on feedback to simulate the negotiation process between experts and the moderator to improve consensus and reach agreed solutions. Both models have been evaluated and their performance has been validated in experiments that compare several configurations of the proposals and baseline techniques. The

results show that the extension of group recommender systems to integrate consensus reaching processes benefits recommendation results, whose performance overcome those of the baseline in the evaluation measures considered. A graphic visualization of the recommendation complements the results of the experiments and shows that the framework for consensus-driven group recommender systems proposed in this chapter delivers recommendations that satisfy individuals.

Chapter 5

Opinion Dynamics-based Group Recommender Systems

5.1 Introduction

Previous chapter proposed a framework for consensus-driven group recommender systems that delivers agreed recommendations to groups of users and proved that the application of consensus reaching processes benefits group recommendation. Consensus towards recommendations has been highlighted as an important aspect in group recommendation, and previous results suggest that applying consensus models in the group recommendation might be a good direction for researching. However, the previous framework does not consider that user's behavior may change when they are part of a group. In these cases, members' preferences might be influenced by the preferences of their peers. Given that this effect happens in several real-world scenarios involving groups, it is necessary to propose models that are aware of opinion influence among group's members.

This chapter studies the application of opinion dynamics models to group recommendation. Opinion dynamics models have been previously applied to study opinion evolution in groups according to various assumptions over the dynamics of the group. Among the various opinion dynamics models in the literature [62, 82, 102, 134], we initially use the DeGroot model [68] because it is suitable to consider influences among members regarding their preferences (see Section

2.5.3). DeGroot's model establishes that individuals' opinions are influenced by others' opinions with a certain weight. Therefore, this social process allows to model such an influence effect in GRSs and its integration in group recommendation is studied in this chapter.

The opinion dynamics process in DeGroot's model has two outcomes: consensus or fragmentation. In the consensus outcome, all opinions converge to the same value and the group's preference over a given item is unique, hence, there is agreement over the final value and the group recommendation is regarded as successful by all members. In the fragmentation outcome, there are more than one final opinion values, hence, group's members do not agree on a preference value over one item, which may bias the recommendation. Previous researches determined that providing a consensus solution benefits recommendations [78], hence, it is clear that consensus among group's members is a desirable feature of the system. Moreover, previous works that seek consensus do not consider influences among group's members [57]. Therefore, ensuring consensus in opinion dynamics group recommender systems might lead to better recommendations.

We aim to study the effect of two aspects in the group recommendation: (i) considering influence of members opinions, and (ii) providing consensus solutions. Hence, we propose two models:

- **Pre-GROD.** It extends DeGroot's model for group recommendation, which allows to consider influences and relationships between members' preferences.

- **GROD**. The consensus conditions are ensured in DeGroot's model applied to group recommendation in order to compute agreed recommendations that are agreed by all members.

The remainder of this chapter is structured as follows. The new GRS framework is developed in Section 5.2. The experiments and results are demonstrated in Section 5.3. Section 5.4 presents a sensitivity analysis of GROD. A summary is

provided in Section 5.5.

5.2 Framework for group recommendation based on opinion dynamics

This section details the new framework for group recommendation based on opinion dynamics. The aim of this framework is to take relationships between members' preferences into account in the group recommendation process. For this, the opinion dynamics model applied needs the initial members' opinion and the relationships between them to drive the change.

In the case of the initial opinions, either the rating value or the rating prediction for each member could be used. However, the usage of the initial rating value would lead to apply a pseudo-user based group recommendation approach and the opinion dynamics model would only influence how the pseudo user is computed. Instead, we focus on recommendation aggregation GRSs because the opinion dynamics model influence on the individual recommendation seems more promising, therefore, the individual prediction value is used as the initial members' opinion.

Regarding the relationships among members, their specific definition is dependent on the aim of the system. We propose to consider similarity of preferences and overlap of experiences, but other features might be considered. The similarity measure is a key part of the opinion dynamics model because it defines the way the relationships matrix is computed. These relationships drive the evolution of individual predictions towards a collective prediction value. Unlike traditional aggregation-based GRSs, the framework is able to flexibilize the process to calculate the group recommendation by means of a matrix of weights between group's members. The general framework for group recommendation based on opinion dynamics is depicted in Figure 5.1, and comprises the following phases:

(1) Compute individual predictions.

(2) Compute the relationships between members' preferences.

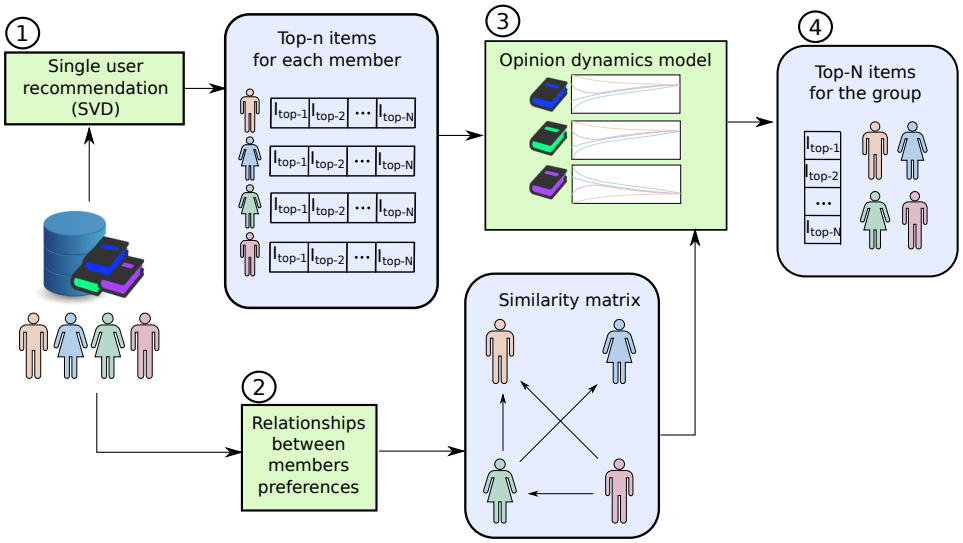


Figure 5.1: Framework for group recommender system based on opinion dynamics.

(3) Apply DeGroot's model for each item to predict the group rating.

(4) Recommend the items with the highest prediction.

Two novel methods that follow the general framework depicted above are described: (i) a group recommender system based on opinion dynamics (Pre-GROD) and (ii) and a group recommender system based on opinion dynamics with consensus (GROD). These approaches combine individual predictions driven by members' preferences relationships. Unlike Pre-GROD, GROD ensures the consensus adding a step that analyses and modifies the weights among members. The remainder of this section further details both proposals.

5.2.1 Group recommender system based on opinion dynamics (Pre-GROD)

Pre-GROD follows the phases stablished in the general framework. In the first phase, individual predictions are computed for the target item i_k. For such a prediction, the stochastic gradient descent singular value decomposition (SVD) RS [126] is used. This prediction method ensures that a prediction value can be computed for any user u_j and item i_k pair, i.e. full coverage, as long as their corresponding rating sets $R_{u_j\bullet}$ and $R_{\bullet i_k}$ are not empty. The group prediction will be derived from these individual predictions, hence, the full coverage requirement over the single user RS ensures that all members are considered in the recommendation.

In the second phase, member's preferences relationships are calculated. Matrix S contains the relationships between members and is used to build the weighting matrix among members, which drive the opinion dynamics process. The specific definition of the similarity used defines how the individual predictions evolve in the opinion dynamics process to obtain the group prediction. In the experiments (see Section 5.3) various definitions for the similarity measure [104, 180] are studied in order to determine their suitability (see Appendix B):

$$s_{u_j,u_k} = similarity(u_j,u_k) \in [0,1], \forall u_j,u_k \in G \subseteq U \tag{5.1}$$

In the third phase, the group predictions are calculated. The similarity matrix S from previous phase is converted in a weighting matrix A using Eq. 5.2:

$$a_{u_j,u_k} = \frac{s_{u_j,u_k}}{\sum\limits_{u_l \in G} s_{u_j,u_l}} \tag{5.2}$$

In the third step the group predictions are computed. Once the weighting matrix A is obtained, individual predictions are combined for each item applying DeGroot's model. To do so, individual predictions are considered to be the initial group's members opinion over the item, and their final value when DeGroot's

model process stabilizes determines the final members' opinion.

$$X_{i_k}^0 = (\tilde{r}_{u_j,i_k})_{(1\times|G|)} \tag{5.3}$$

$$\tilde{r}_{G,i_k} = \lim_{t\to\infty} A^t X_{i_k}^0 \tag{5.4}$$

When the process stabilizes, the group prediction for item i_k is obtained averaging the final members' opinion over the target item $\tilde{r}_{G,i_k}$. The same process is applied to each item, thus, all the group predictions for each considered item are obtained.

In the fourth phase, the recommendation is composed selecting the items with highest group rating prediction.

The Pre-GROD proposal performance is exemplified as follows. Table 5.1 shows the ratings of a group of users composed of five members. Table 5.2 shows the pairwise similarities of all members computed with Pearson's correlation coefficient and keeping only positive values. Table 5.3 shows the weighting matrix, whose normalized left eigenvector associated to eigenvalue 1 is $(0.22, 0.25, 0.20, 0.19, 0.14)$. If we consider the following initial opinions $\tilde{r}_{m_1,i_{11}} = 3.5$, $\tilde{r}_{m_2,i_{11}} = 5.0$, $\tilde{r}_{m_3,i_{11}} = 5.0$, $\tilde{r}_{m_4,i_{11}} = 2.5$, $\tilde{r}_{m_5,i_{11}} = 2.8$, which are produced by the individual recommender system, then the opinion dynamics model ends with all members having the same final opinion over target item i_{11}, which is $\tilde{r}_{G,i_{11}} = 3.88$.

$r_{u_j i_k}$	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}
m_1		2					1	0,5		
m_2	2,5	4	2,5		0,5		2			
m_3	2,5	2		1	0,5	4		2,5	4,5	2
m_4	3,5		1,5	1			1			
m_5					4		4	2	0,5	3,5

Table 5.1: Ratings considered for the example of application of Pre-GROD.

$s_{m_j m_k}$	m_1	m_2	m_3	m_4	m_5
m_1	1	1	0	0	1
m_2	1	1	0,78	0,65	0
m_3	0	0,78	1	1	0
m_4	0	0,65	1	1	0
m_5	1	0	0	0	1

Table 5.2: Similarity matrix extracted from the ratings in Table 5.1.

$a_{m_j m_k}$	m_1	m_2	m_3	m_4	m_5
m_1	0.33	0.33	0	0	0.33
m_2	0.29	0.29	0.23	0.19	0
m_3	0	0.27	0.36	0.36	0
m_4	0	0.25	0.38	0.38	0
m_5	0.50	0	0	0	0.50

Table 5.3: Weights matrix derived from similarity matrix

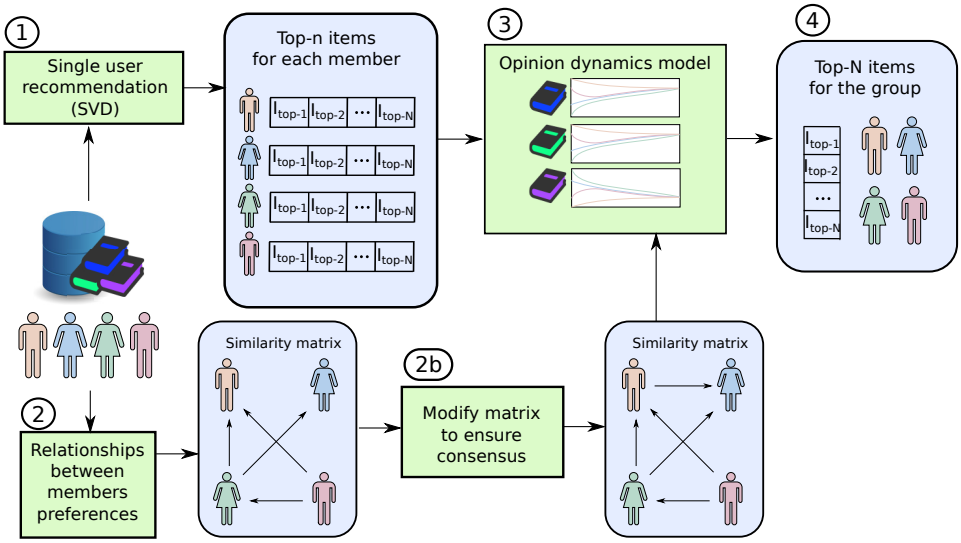


Figure 5.2: Group recommender system based on opinion dynamics with consensus (GROD).

5.2.2 Group recommendation based on opinion dynamics with consensus (GROD)

GROD model is described in this section, and it shares the general scheme with Pre-GROD. The difference is that GROD includes an additional step in the second phase to ensure that relationships among members lead to consensus, because previous chapter proved that consensus provides better group recommendations. Hence, the additional step analyses the similarity matrix to detect those cases that do not lead to consensus, which diminishes members satisfaction towards the recommendation, and corrects it. The general scheme of the framework is modified to include such an additional step (see Figure 5.2)

The *step 2b* of Figure 5.2 analyses the relationships matrix S and, in the case it does not lead to consensus, it is modified to do so. After this modification, the relationships matrix S' is used to determine the weighting matrix A' and the opinion dynamics process can be finished with the certainty that a consensus opinion is reached.

In cases where the relationship matrix S does not lead to consensus, the left eigenvector associated to eigenvalue 1 of weighting matrix A is not unique. Their multiplicity determines how many opinion subgroups are present, and each associated eigenvectors determines how opinions evolve within each subgroup and which users belong to each of them.

In order to ensure consensus, these opinion subgroups need to be connected, therefore it is needed to add at least $q-1$ relationships to connect them, where q is the number of opinion subgroups [70]. There are many options for adding the opinion subgroups. In order to decide which relationships are added, GROD computes a score for each missing relationship and then selects the one with the highest score. GROD uses the number of ratings of the target subgroup for such a score in order to consider opinion subgroups with a more defined taste. This way, a directional edge with degree of relationship of 1 is added to the relationships matrix S

$$arg\ max_{u_j,u_k}(score(u_j,u_k)) \tag{5.5}$$

$$score(u_j,u_k) = |R_{G^{u_k}\bullet}| = \sum_{u_l \in G^{u_k}} |R_{u_l\bullet}| \tag{5.6}$$

where G^{u_k} is composed of group G members that are in the opinion subgroup of member u_k, and $R_{G^{u_k}\bullet}$ is set of ratings of these members. The opinion subgroup of member u_k can be determined taking the positive values of the eigenvector associated to u_k opinion subgroup.

This procedure is repeated until matrix S' leads to consensus. On each iteration, the weighting matrix A' changes and the number of subgroups, which is determined

by the number of eigenvalues, is reduced by one. At the end of this process, the opinion subgroup is equal to the whole group, and the consensus is guaranteed.

5.3 Experimentation and evaluation

As aforementioned, both proposed models test different improvements to the group recommendation, therefore, two experiments were performed. The first one evaluates Pre-GROD for group recommendation to test whether the consideration of group relationships improves group recommendation and determine the best definition for the relationships. The second one evaluates GROD for those groups that do not reach consensus to test whether ensuring consensus improves group recommendation.

The remainder of this section is structured as follows. First, the techniques compared on both experiments are described. After that, the datasets and methods for processing them are detailed. Later, the evaluation measures are defined. Eventually, the results for both experiments are shown and analyzed.

5.3.1 Techniques compared

In both experiments, the baseline method is a GRS based on recommendation aggregation with Mean. The stochastic gradient descent method [126] with the configuration shown in Table 5.4 is used to compute individual predictions.

Table 5.4: Parameters for the individual predictor

Parameter	Value
Number of features	20
Iterations per feature	20
Learning rate (γ)	0.01
Large values penalty (λ)	0.02

In the first experiment, Pre-GROD is evaluated with five similarity measures between users (see Appendix B), in order to compare various relationships notions and test their performance:

- *Pre-GROD Cosine*: Cosine coefficient [203].
- *Pre-GROD Relevance(30)*: Relevance factor with r=30 [104].
- *Pre-GROD Cond. Prob.*: Conditional probability [180].
- *Pre-GROD Asym. Cos.*: Asymmetric cosine [180].

In the second experiment, GROD is evaluated to determine whether ensuring consensus has a positive impact in groups that initially do not reach consensus (see Appendix A). Thus, three techniques are compared: (i) the baseline, (ii) our proposed GRS based on opinion dynamics (Pre-GROD) and (iii) our proposed GRS based on opinion dynamics with consensus (GROD).

In order to discard any effect of the individual ratings predictor, all techniques compared have the same individual predictor as the baseline.

5.3.2 Datasets

The datasets used in the experiments are shown in Table 5.5. These datasets do not contain information about groups, hence, as aforementioned in the experimental section of previous chapter, there are different methods of forming groups. In these experiments, random groups are formed [228] to evaluate techniques for occasional groups. The sizes of the groups formed ranged from 1 to 10 members.

Table 5.5: Main features of the datasets used in the experiments over the framework of group recommendation based on opinion dynamics.

Dataset	Users	Items	Ratings	Sparsity
MovieLens-100k	943	1,682	100,000	93.69%
MovieLens-1m	6,040	3,706	1,000,209	95.53%

5.3.3 Evaluation measures

Two widely used evaluation metrics are considered in the experiment for evaluating the accuracy of the two GRS proposals: mean absolute error (MAE), and root mean square error (RMSE) [95, 103]:

- *MAE*: Reflects the average difference between the system prediction and the true rating value.

$$MAE = \frac{1}{|R|} \sum_{r_{u_j,i_k} \in R} |r_{u_j,i_k} - \tilde{r}_{u_j,i_k}| \tag{5.7}$$

  where $\tilde{r}_{u_j,i_k}$ is the GRS prediction and r_{u_j,i_k} is the true rating value in the test dataset. Notice that in GRSs all members receive the same rating prediction, but their ratings are different.

- *RMSE*: similarly, RMSE measures the prediction error, but it gives more importance to larger errors.

$$RMSE = \sqrt{\frac{1}{|R|} \sum_{r_{u_j,i_k} \in R} (r_{u_j,i_k} - \tilde{r}_{u_j,i_k})^2} \tag{5.8}$$

The GRSs performance is better when their values are lower, given that both of them measure prediction error. The dataset is split in training and test set performing a 20 executions 5-cross fold validation. The groups and the training-test partitions were different in each execution. Each execution results were averaged to obtain the final value for each GRS.

5.3.4 Experiment 1: Pre-GROD in the general case

The aim of this experiment is to evaluate Pre-GROD with various similarity measure definitions to determine their performance, the suitability of this proposal, and compare it with the baseline.

Figure 5.3 shows the results of the GRSs on MovieLens-100k. Within it, Figure 5.3a shows the results for MAE and Figure 5.3b shows the results for RMSE. In the same way, Tables 5.6 and 5.7 show the results for MAE and RMSE, respectively. Both in the figures and in the tables, the results are stratified by group size, ranging from 1 to 10 members.

The results clearly show that Pre-GROD with the Asym. Cos. similarity reached the best performance over the baseline. The relative improvement was

Table 5.6: MAE for the evaluated GRSs in MovieLens 100k

GRS		Group size 1	2	3	4	5	6	7	8	9	10
	Baseline	0.7418	0.7816	0.7931	0.8004	0.8044	0.8050	0.8070	0.8077	0.8082	0.8092
Pre-GROD	Cosine	0.7418	0.7816	0.7927	0.7999	0.8038	0.8045	0.8065	0.8072	0.8077	0.8088
Pre-GROD	Relevance(30)	0.7418	0.7790	0.7887	0.7956	0.7995	0.8004	0.8028	0.8037	0.8045	0.8059
Pre-GROD	Cond. prob.	0.7418	0.7685	0.7802	0.7890	0.7944	0.7966	0.7998	0.8012	**0.8024**	**0.8042**
Pre-GROD	Asym. Cos.	0.7418	**0.7684**	**0.7800**	**0.7887**	**0.7941**	**0.7964**	**0.7997**	**0.8011**	**0.8024**	0.8043

Table 5.7: RMSE for the evaluated GRSs in MovieLens 100k

GRS		Group size 1	2	3	4	5	6	7	8	9	10
	Baseline	0.9442	0.9827	0.9954	1.0044	1.0091	1.0103	1.0127	1.0137	1.0144	1.0157
Pre-GROD	Cosine	0.9442	0.9827	0.9951	1.0040	1.0086	1.0099	1.0123	1.0133	1.0141	1.0154
Pre-GROD	Relevance(30)	0.9442	0.9798	0.9904	0.9985	1.0029	1.0043	1.0070	1.0081	1.0091	1.0107
Pre-GROD	Cond. prob.	0.9442	**0.9705**	**0.9821**	**0.9912**	**0.9968**	**0.9991**	**1.0024**	1.0040	1.0054	1.0073
Pre-GROD	Asym. Cos.	0.9442	0.9707	0.9822	0.9913	**0.9968**	**0.9991**	**1.0024**	**1.0039**	**1.0053**	**1.0072**

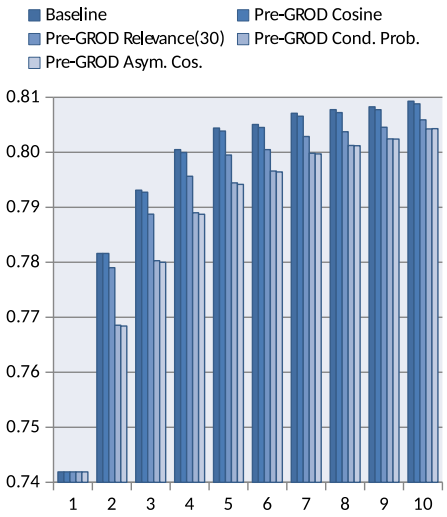
1% across all group sizes on average. Improvement was greater in smaller group sizes. Relative improvement in MAE was 1.72%, 1.68% and 1.49% for groups of sizes 2, 3 and 4, respectively. In RMSE, it was 1.24%, 1.35% and 1.32%, respectively. The results of both Pre-GROD Cosine and Pre-GROD Cond. Prob. were overcome by those of Pre-GROD Asym. Cos, which indicates that accounting for both exposure to the same experiences, and correlation between the satisfaction on these experiences improves performance.

Asymmetric measures, such as Pre-GROD Relevance(30), Pre-GROD Cond. Prob. and GROD Asym. Cos. showed better results than symmetric ones considering Pre-GROD's performance with various ways to determine the relationships between members' preferences. Asymmetric measures consistently obtained better results than Pre-GROD Cosine. Symmetric measures obtained results similar to the baseline. This fact indicates that the system achieves better results when asymmetric similarities are considered, that might be motivated by the inherent asymmetry of relationships between people.

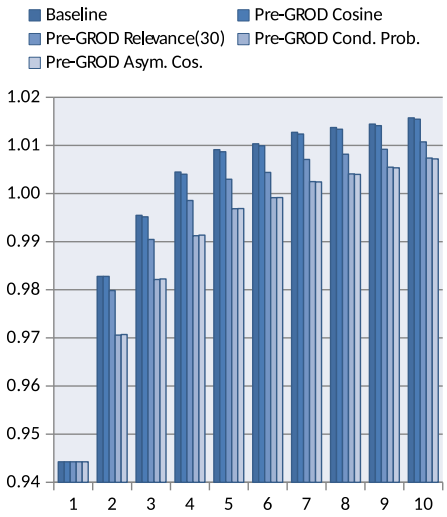
Finally, similar behavior is observed, in general, comparing the results on MAE and RMSE for the different GRSs. This behavior means that the balance between small and large errors is similar. Cond. Prob. obtained better results for RMSE and

Figure 5.3: Results for the evaluated GRSs in MovieLens 100k.

(a) MAE by group size.



(b) RMSE by group size.



Asym. Cos. for MAE, which is an interesting observation. This result indicates that the errors of *Cond. Prob.* are smaller than Asym. Cos. ones.

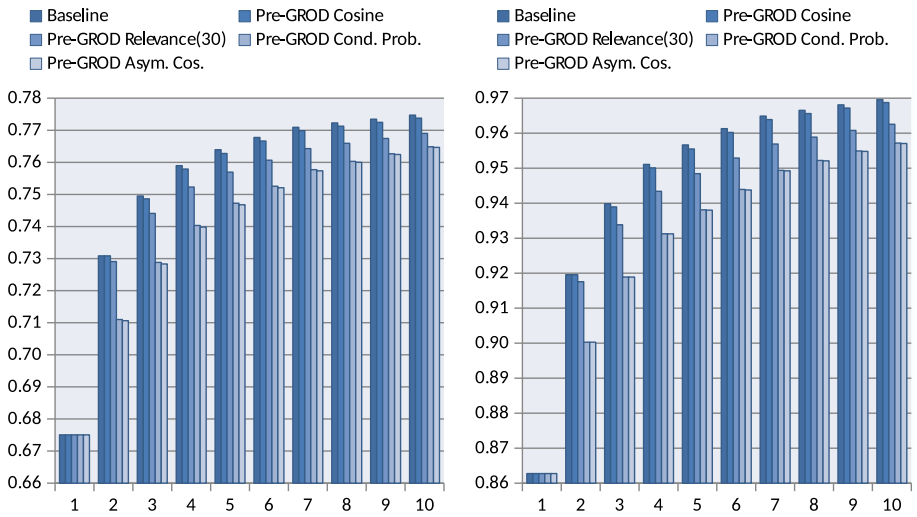
Similarly, the results for MovieLens-1M are shown in Figure 5.4, where Figures 5.4a and 5.4b show the MAE and RMSE, respectively. The results have been stratified by group size ranging from 1 to 10 members.

Overall, similar behavior to MovieLens-100k was achieved. Although, the results obtained in MovieLens-1M were better than MovieLens-100k. This behavior is motivated by better quality of the model generated by the RS due to MovieLens-1M dataset containing more ratings, which increases the final accuracy of the GRS and decreases the prediction error.

Mainly, the proposed method with the Asym. Cos. measure achieved the best performance compared to the other approaches. The average relative improvement of roughly 2% over the baseline across all group sizes evaluated was better, and it achieved a greater improvement than MovieLens-100k. For smaller groups, the improvement was also greater. The relative improvement in RMSE was 2.14%, 2.28% and 2.13% for groups sizes 2, 3 and 4, respectively. In MAE, it was 2.85%,

Figure 5.4: Results for the evaluated GRSs in MovieLens 1M.

(a) MAE by group size. (b) RMSE by group size.



2.91% and 2.59% for the same group sizes.

These results support the statement that Pre-GROD Asym. Cos. improves the results in general recommendation cases, for random groups, compared to the baseline, with an improvement of 1% for MovieLens-100k and 2% in MovieLens-1M. Moreover, Asymmetric Cosine is the best configuration to compute the relationships between member preferences in the proposal.

5.3.5 Experiment 2: GROD in groups without consensus

The aim of this experiment is to evaluate the improvements achieved ensuring consensus. To do this, results on groups without opinion leaders are analyzed (see Appendix A). This way, the effect of correcting the weighting matrix is isolated and it is possible to measure the improvement that it provides. Three GRSs are compared in this experiment: (i) the baseline, a GRS based on recommendation aggregation; (ii) Pre-GROD, the proposal without consensus; and (iii) GROD, the proposal ensuring consensus.

Figure 5.5 shows the results for MovieLens-100k. MAE is shown in Figure 5.5a and RMSE is shown Figure 5.5b. The results are stratified by group size,

Table 5.8: MAE for the evaluated GRSs in MovieLens 1m

		Group size									
GRS		1	2	3	4	5	6	7	8	9	10
	Baseline	0.6751	0.7309	0.7495	0.7590	0.7639	0.7678	0.7709	0.7723	0.7735	0.7747
Pre-GROD	Cosine	0.6751	0.7309	0.7486	0.7579	0.7628	0.7666	0.7699	0.7713	0.7725	0.7738
Pre-GROD	Relevance(30)	0.6751	0.7291	0.7441	0.7523	0.7569	0.7607	0.7643	0.7659	0.7674	0.7690
Pre-GROD	Cond. prob.	0.6751	0.7110	0.7288	0.7403	0.7473	0.7526	0.7577	0.7603	0.7627	0.7648
Pre-GROD	Asym. Cos.	0.6751	**0.7106**	**0.7283**	**0.7398**	**0.7468**	**0.7521**	**0.7574**	**0.7600**	**0.7625**	**0.7647**

Table 5.9: RMSE for the evaluated GRSs in MovieLens 1m

		Group size									
GRS		1	2	3	4	5	6	7	8	9	10
	Baseline	0.8628	0.9195	0.9398	0.9511	0.9566	0.9613	0.9649	0.9665	0.9681	0.9696
Pre-GROD	Cosine	0.8628	0.9195	0.9390	0.9501	0.9555	0.9602	0.9639	0.9656	0.9672	0.9687
Pre-GROD	Relevance(30)	0.8628	0.9176	0.9338	0.9434	0.9484	0.9529	0.9569	0.9589	0.9608	0.9625
Pre-GROD	Cond. prob.	0.8628	**0.9003**	**0.9189**	0.9313	0.9381	0.9439	0.9494	0.9522	0.9549	0.9572
Pre-GROD	Asym. Cos.	0.8628	**0.9003**	**0.9189**	**0.9312**	**0.9380**	**0.9438**	**0.9493**	**0.9521**	**0.9548**	**0.9571**

similarly to previous experiment, to determine the improvement of the proposal for each case.

Table 5.10: MAE for the evaluated GRSs in MovieLens 100k

	Group size									
GRS	1	2	3	4	5	6	7	8	9	10
Baseline	0.7792	0.8091	0.8145	0.8191	0.8205	0.8213	0.8232	0.8255	0.8273	0.8285
Pre-GROD	0.7792	0.8091	0.8088	0.8133	0.8152	0.8166	0.8191	0.8215	0.8234	0.8247
GROD	0.7792	**0.7820**	**0.7999**	**0.8092**	**0.8128**	**0.8152**	**0.8183**	**0.8211**	**0.8232**	**0.8246**

Table 5.11: RMSE for the evaluated GRSs in MovieLens 100k

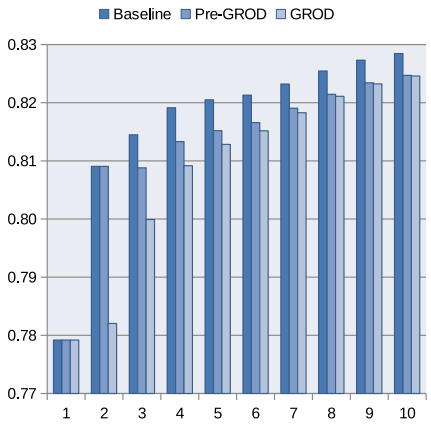
	Group size									
GRS	1	2	3	4	5	6	7	8	9	10
Baseline	1.0009	1.0199	1.0262	1.0308	1.0317	1.0323	1.0342	1.0364	1.0381	1.0390
Pre-GROD	1.0009	1.0199	1.0200	1.0240	1.0253	1.0263	1.0287	**1.0310**	**1.0329**	**1.0341**
GROD	1.0009	**1.0019**	**1.0141**	**1.0215**	**1.0239**	**1.0257**	**1.0286**	1.0312	1.0332	1.0345

The results show that, on groups without consensus and for all group sizes, GROD improved the results over the baseline. Moreover, Pre-GROD results were improved by GROD for most group sizes evaluated. Therefore, in this scenario, correction improves recommendations, as suggested by these results.

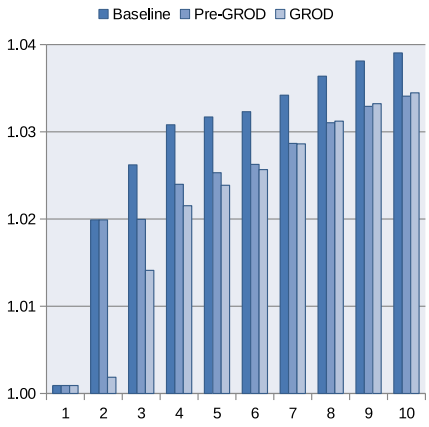
Furthermore, GROD's improvement is greater in smaller groups as shown for both measures. This improvement is achieved for all groups in MAE and for group

Figure 5.5: Results for the evaluated GRSs in MovieLens-100k.

(a) MAE by group size.



(b) RMSE by group size.



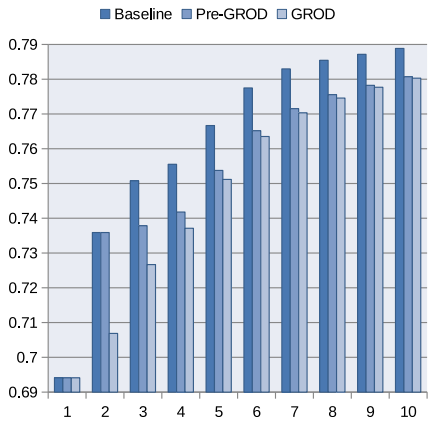
sizes lower than 7 in RMSE, which indicates that GROD is suitable for recommending items to small groups.

An interesting fact is that the proposal without correction obtained exactly the same MAE and RMSE as the baseline for groups of two members. The prediction for groups without consensus is computed with the average of the final opinions, hence, the recommendation was the same than the output of the baseline. The weights in groups of size two that do not reach consensus are necessarily zero, therefore, their opinion does not change in the opinion dynamics process. Hence, the prediction for these groups is always the average of individual predictions.

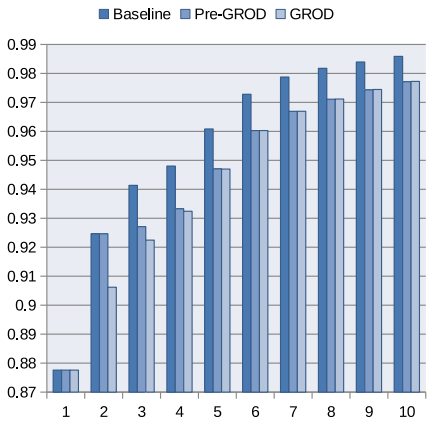
In groups of size 3 and more the recommendations obtained by the baseline and GROD are not the same, given that individual predictions are modified in the opinion dynamics process. In groups that do not reach consensus there are, necessarily, at least two opinion groups. In the specific cases in which the number of opinion groups is equal to the number of group's members there is no change of opinion after the opinion dynamics process and the average of the individual opinions is returned. If the number of opinion groups is strictly less than the number of members, then there is necessarily opinion change for one or more group's members, and the group prediction may differ from the average of individual opinions.

Figure 5.6: Results for the evaluated GRSs in MovieLens-1M.

(a) MAE by group size.



(b) RMSE by group size.



Similarly, Figure 5.6 shows the results for MovieLens-1M. Specifically, Figure 5.6a shows the MAE and Figure 5.6b shows the RMSE. The results are stratified by group size to determine the improvement of the proposal for each case. In general, the results show a similar behavior to the behavior observed for MovieLens-100k.

Table 5.12: MAE for the evaluated GRSs in MovieLens 1M

	Group size									
GRS	1	2	3	4	5	6	7	8	9	10
Baseline	0.6941	0.7359	0.7508	0.7555	0.7667	0.7775	0.7830	0.7854	0.7872	0.7889
Pre-GROD	0.6941	0.7359	0.7378	0.7418	0.7538	0.7652	0.7715	0.7756	0.7782	0.7807
GROD	0.6941	**0.7069**	**0.7267**	**0.7371**	**0.7512**	**0.7635**	**0.7703**	**0.7746**	**0.7777**	**0.7803**

Table 5.13: RMSE for the evaluated GRSs in MovieLens 1M

	Group size									
GRS	1	2	3	4	5	6	7	8	9	10
Baseline	0.8776	0.9246	0.9414	0.9480	0.9608	0.9728	0.9787	0.9818	0.9839	0.9859
Pre-GROD	0.8776	0.9246	0.9271	0.9333	0.9471	**0.9603**	**0.9669**	**0.9711**	**0.9743**	**0.9771**
GROD	0.8776	**0.9062**	**0.9225**	**0.9324**	**0.9470**	**0.9603**	**0.9669**	0.9712	0.9745	0.9772

In general, it can be determined that results are improved for most group sizes and in both evaluation measures with the correction of weights to ensure consensus (GROD). For smaller groups, the improvement is greater, which makes GROD useful, specifically, for recommending to groups composed of seven members or

less.

5.3.6 Computational complexity

The evaluated approaches were analyzed regarding their computational complexity. Table 5.14 details the computational complexity of each task within the compared approaches, and Table 5.15 shows the computational complexity of each approach. In these tables, k denotes the number of factors in the SVD model, $|G|$ is the group size, $|I|$ is the number of items, $|R_G^{max}|$ is the largest amount of ratings of a member, and q is the number of iterations that the DeGroot model needs to converge.

Table 5.14: Computational complexity of proposal tasks.

Task	Complexity	Baseline	Pre-GROD	GROD
Individual predictions	$\mathcal{O}(\|G\|\|I\|k)$	✓	✓	✓
Similarity matrix	$\mathcal{O}(\|G\|^2\|R_G^{max}\|)$		✓	✓
Correct similarity matrix	$\mathcal{O}(\|G\|^3)$			✓
Group prediction	$\mathcal{O}(\|I\|)$	✓	✓	✓

Table 5.15: Computational complexity of evaluated approaches tasks.

GRS approach	Computational complexity
Baseline	$\mathcal{O}(max(\|G\|\|I\|k \ , \ \|I\| \ ))$
Pre-GROD	$\mathcal{O}(max(\|G\|\|I\|k \ , \ \|G\|^2\|R_G^{max}\| \ , \ \|I\| \ ))$
GROD	$\mathcal{O}(max(\|G\|\|I\|k \ , \ \|G\|^2\|R_G^{max}\| \ , \ \|G\|^3 \ , \ \|I\| \ ))$

The group size is much smaller than the maximum number of ratings that a member of such a group has in a real GRS, therefore, the correction of the relationship matrix is finished faster than the computation of the relationship matrix itself. Both task are performed in GROD, while Pre-GROD does not perform the correction, which makes GROD to be computationally more expensive, but it is justified by the better performance obtained in terms of accuracy.

5.4 Sensitivity analysis

The recommendation process is resource consuming in large scale systems. Therefore, to consider this scenario and make our proposal less resource demanding, we

aim to analyze the change of the output, i.e., recommendation; when the inputs, i.e., rating or group's members; change. A sensitive analysis is performed to determine the cases where the recommendation delivered to a group might be the same without incurring in a large error. This behavior saves computations, which reduces the requirements of deploying such a system because it reduces the system load.

As aforementioned, two main inputs exist in group recommendation scenarios: group's members and their ratings. Two separate sensitive analysis are performed to consider changes in each of them. The first sensitive analysis studies the changes in the recommendation when a user leaves the group. The second sensitive analysis focuses on changes in members' preferences. Specifically, it analyses the changes when one member changes their preferences. The aim of both sensitive analyses is to quantify the amount of changes in the recommendation that a change in group's members and their ratings does, in order to determine when a major change in recommendations happens, and therefore, a recalculation of the recommendations must be done.

Hence, the changes in the recommendation are measured in both sensitive analyses. The main feature of the recommendation is the sorting of the recommended items, because small changes in the rating prediction for individuals or for the group may not be enough to change the sorting of the items in the recommendation. This situation is exemplified as follows. In case A the GRS recommends products 1, 2 and 3 with prediction values of 3.9, 3.8 and 3.7, respectively. In case B, the GRS recommends products 1, 2 and 3 with prediction values of 5, 4.9 and 4.8, respectively. And, in case C the GRS returns the items ordered as 3, 2, 1 with prediction values of 3.9, 3.8 and 3.7, respectively. As it can be noticed, predictions change for case B, but the sorting of items does not change. In case C the raking of the items changes, even though the changes in rating prediction are smaller as compared to the changes of case B. This example lead us to ignore the prediction value and focus on item ranking for sensitive analyses.

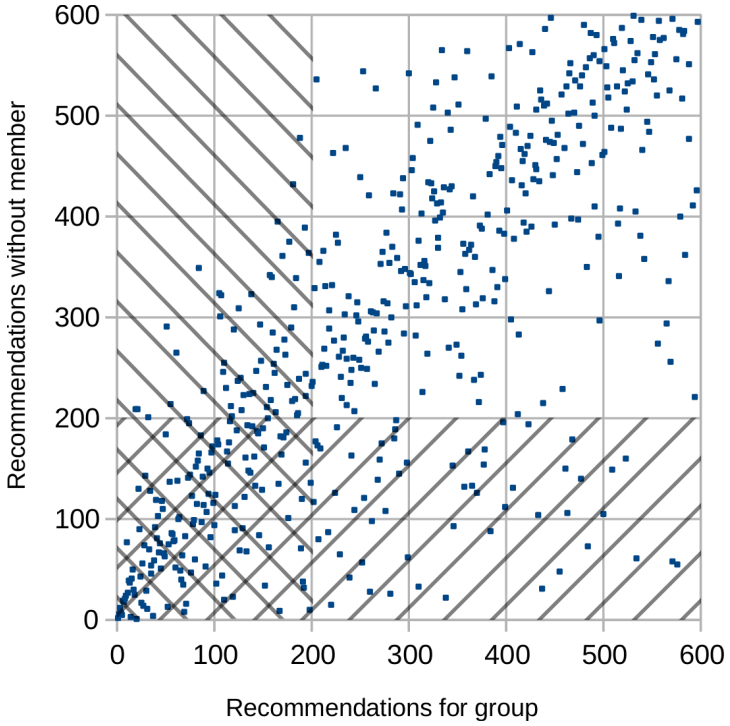


Figure 5.7: Changes in the item ranking for group [126, 595, 607, 619, 648] and when user 126 leaves.

The changes in the recommendation are measured with the following measures:

- *Intersection@size*: Quantifies the differences in recommendations accounting for the number of items in the intersection among the number of recommended items.

- *Spearman@size*: Spearman rank correlation is used quantify the similarity between the rankings of two different recommendations.

Both measures are exemplified in Figure 5.7, where an example of two recommendations is shown. Items ranking is shown in the scatter plot for the complete group and for the group without member 126. The items recommended to the complete group are depicted with the rectangle filled with descending diagonal lines. The recommendations of the group without member 126 are depicted in the rectangle with ascending diagonal lines. The intersection of both rectangles, with both

ascending and descending diagonal lines, contains items recommended in both cases with a recommendation list size of 200. The square without lines contains items that are not recommended in either case. The ratio of items in the common square of Figure 5.7 is the Intersection@200 and the ranking correlation of these items is analyzed with Spearman@200.

5.4.1 Member elimination

The influence of a member that leaves the group is analyzed in this sensitive analysis. We aim to determine the exact point at which is better to recompute the recommendations in order to avoid incurring in a larger error, or when it is possible to deliver the same recommendations to save computational resources.

Figure 5.8 and 5.9 show the results of the Intersection@Size and Spearman@Size for MovieLens-100k. In order to compare the influence of the member's absence across group sizes, the results are stratified. Logarithmic scale is used for the recommendation size in order to highlight changes at the beginning of the list, which are perceived by users.

The absence of a member produces greater changes in smaller groups, as the results reveal. The fewer members a group has, the larger influence of a user's absence has, as it is expected due to the major impact of a single profile among the remaining. For large groups, changes in the recommendation tend to diminish.

The results for Intersection@size show that the variability in the recommendation list when inputs change is related to the size of the recommendation list. Figure 5.8 shows that recommendations for group sizes between 10 and 20 have an Intersection@size lower than for groups of sizes between 20 and 30. This fact is an evidence of a user leaving the group has more impact in the former.

Figure 5.10 shows the results of executions for each group in order to focus on the impact of the member that is leaving the group given their characteristics. Recommendations lists of size 5 are checked for groups of size between 2 and 10 on MovieLens-100k. The cases shown are sorted by the number of ratings of

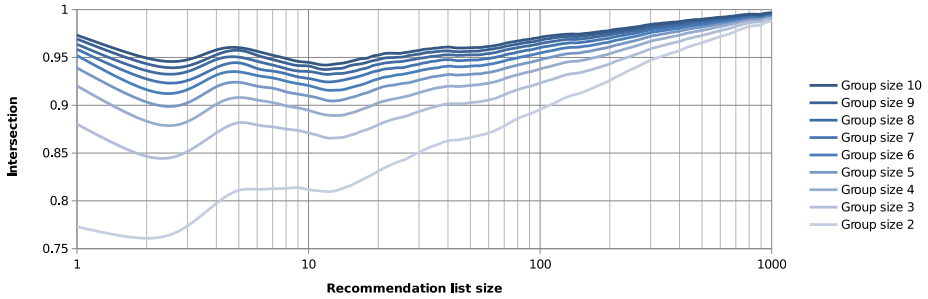


Figure 5.8: Results of Intersection@Size for groups of size 2 to 10 to over the recommendation list.

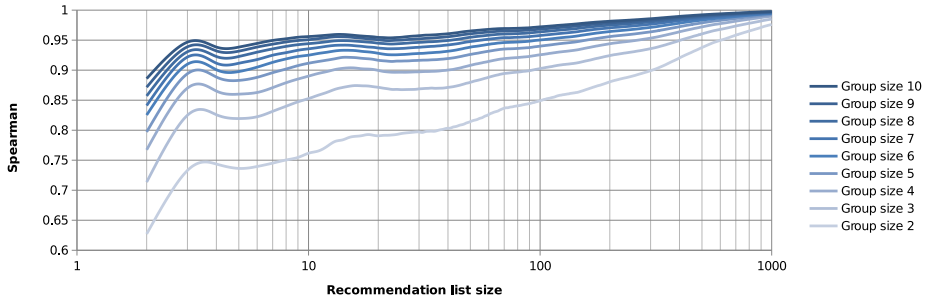
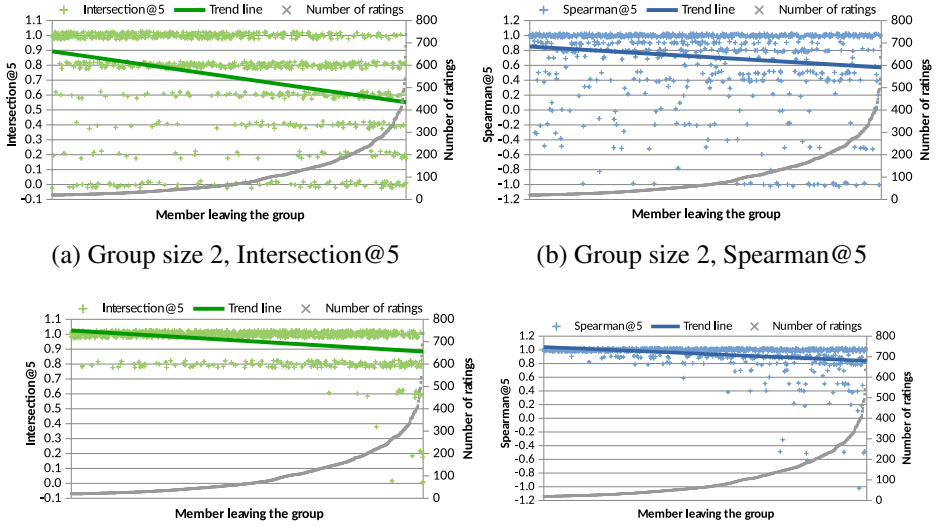


Figure 5.9: Results of Spearman@Size for groups of size 2 to 10 to over the recommendation list.

the user leaving the group. This sorting is made to highlight that the number of ratings of the user leaving influences the impact of that user's contribution to the final value. A jitter value is added to Intersection@5 and Spearman@5 in order to reduce overlap in the data and make the results clearer.

Figure 5.10 includes a trend-line in each sub-figure to highlight that there is a larger impact in the recommendation when the member leaving the group has more ratings. This tendency is the same across all group sizes. Only groups sizes 2 and 10 are shown for the sake of simplicity. In each of these cases, there is an Intersection@5 and a Spearman@5 value for a user leaving the group. The data shows that when the profile of the user abandoning the group has more ratings, its leaving produces larger changes in the outputs. This effect might happen because

Figure 5.10: Changes in the recommendation when member leaves the group sorted by number of ratings.



(a) Group size 2, Intersection@5

(b) Group size 2, Spearman@5

(c) Group size 10, Intersection@5

(d) Group size 10, Spearman@5

users have greater probability of being a leader when they have more ratings.

The conclusions of this sensitive analysis are two:

1. For small groups there is a larger impact on the recommendation when a user leaves.

2. The larger the rating profile the user leaving has, the larger impact in the recommendation its absence has

With the evidence shown in previous results we can conclude that for groups of size five and smaller, the recommendation should always be recomputed when a user leaves. For the remaining group sizes, it is suggested to update recommendations when the user leaving the group has 220 ratings or more. The exact group size and number of ratings were determined considering Intersection@size values that are above 0.9.

5.4.2 Rating variation

In group recommendation, another source of variability in the recommendation is members' ratings. Hence, the impact of a member changing their ratings is measured in this sensitive analysis. The aim with this measurement is to suggest a point where the recommendation should be updated to avoid large errors and when its possible to save computational resources delivering the same recommendation to the group.

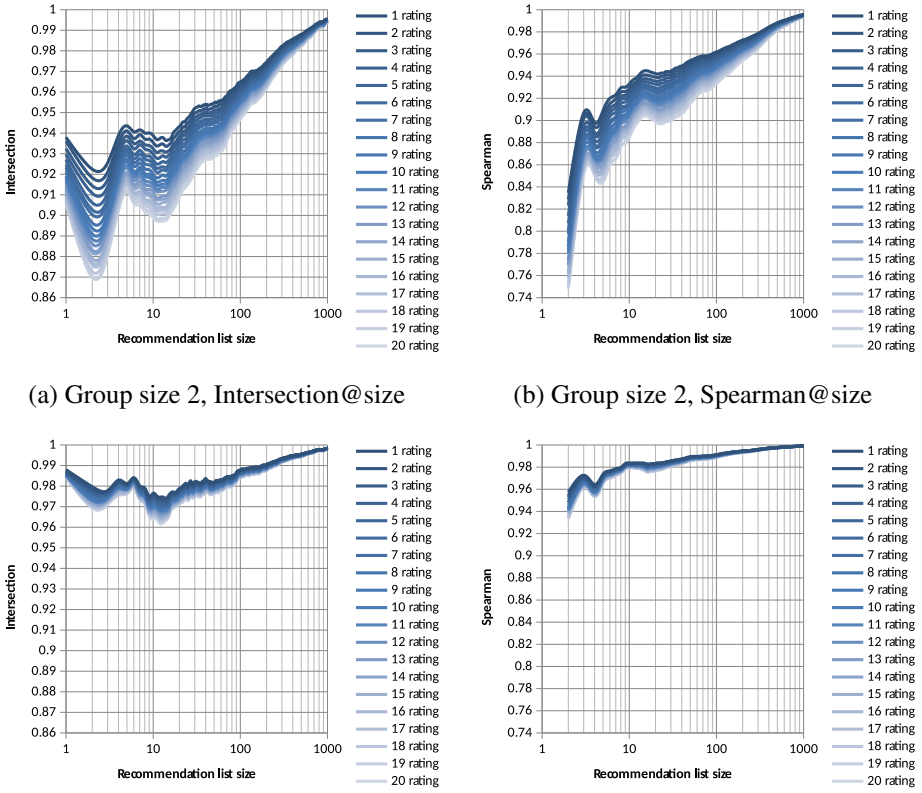
With such an aim, one member of the group changes certain number of ratings ranging between 1 and 20, to a new random value. In this sensitive analysis, the group sizes have also been analyzed. The number of variables that we analyze makes results spread across four dimensions. For the sake of clarity, Figure 5.11 only shows the results for group sizes 2 and 10.

The results reveal that there is a greater change in the recommendation for smaller groups. This behavior is similar to the detected in previous sensitive analysis, the smaller the group the more impact a change in a member's profile has in the recommendation due to the greater weight of a single profile in the recommendation process.

The results also reveal that there is a greater impact on the recommendation when more ratings are modified in a member profile. It is important to consider that the modification of ratings in a single profile only changes the SVD profile, and thus the recommendations, of the user whose ratings are modified, while the SVD profile for the remaining members is the same and hence their individual recommendations. The changes in the recommendation make the opinion dynamics process evolve differently. Hence, if more ratings are modified, then the initial opinion for such user is more different, and the opinion dynamics process evolves to a much more different value. This behavior is the same across all group sizes, with the difference that the more users a group has, the more ratings need to be changed due to the less impact of the changed opinion in the process.

This sensitive analysis results suggests that recommendations are not to be up-

Figure 5.11: Changes in the recommendation when member changes ratings.



(a) Group size 2, Intersection@size

(b) Group size 2, Spearman@size

(c) Group size 10, Intersection@size

(d) Group size 10, Spearman@size

dated when a single user changes their ratings because the impact of the change is not large enough in all cases. In order to determine this suggestion, we focused on cases with the larger impact, this is, in groups of size two (see Figure 5.11a). For groups of size three and larger the Intersection@size is always above 0.9. For groups of size two, the Intersection@size is always above 0.9 but drop below 0.9 only in the case of a user modifying six ratings or more.

5.5 Summary

This chapter presents a framework to extend opinion dynamics and apply it to group recommender systems. This framework improves how individual preferences are aggregated by the GRS considering relationships between members'

preferences in such aggregation, which is performed applying DeGroot' opinion dynamics model. Within this model, consensus can be achieved adding a step to the general framework, which ensures consensus and makes them agreed by all members. Moreover, recommendations agreed by all members are delivered ensuring consensus in the recommendation, which is accomplished adding a step with such an aim to the general framework. The performance of the proposal is evaluated in two experiments. The first experiment compares Pre-GROD with the baseline and evaluates several similarity measures to build the relationships' matrix, and it shows that the proposed framework improves results as compared to the baseline and that the best results were achieved by asymmetric similarity measures. The second experiment evaluates the impact of ensuring consensus in groups where DeGroot's model alone does not reaches consensus measuring the improvement in recommendation accuracy. The results show that ensuring consensus improves results as compared to the baseline and the proposal without ensuring consensus. Two sensitive analysis that complete the experiment are performed, which aim to decide under what circumstances it is possible to deliver the same recommendation or the cases when a change in the inputs should trigger an update in the recommendations. The first sensitive analysis evaluates the impact of a user leaving the group, and the results show that for large groups it is not necessary to update the recommendation, while for smaller groups it is only necessary to update the recommendation when the user leaving the group has a large preference profile. The second sensitive analysis evaluates the impact of a user changing their preferences, and the results show that a large amount of ratings is needed to be changed by an user to require a recommendation update due to large changes in the group recommendation.

Chapter 6

Natural noise management in group recommender systems

6.1 Introduction

Previous chapter proposed GROD, a framework for considering relationships between members' preferences when aggregating individual recommendations and ensures consensus within this process to improve recommendation. However, the performance of recommender systems is always subject to the quality of the ratings that they use as inputs for their computations. This issue was pointed out by Herlocker et al. [103] and named it as the *magic barrier*, which is a lower bound to the prediction error that a recommender system can achieve due to inconsistencies or noise in users ratings [30]. These inconsistencies may result in a loss of accuracy that users might perceive reducing their satisfaction towards the recommendation. Although this effect has been studied for individual recommender systems through the management of natural noise, it is needed to study it in group recommender systems.

This chapter studies the extension of natural noise management techniques to integrate them in group recommender systems. Previous works proved that natural noise affects individual preferences and biases individual recommendations. Given that many group recommender systems are based on aggregation of individual recommendations and almost all of them use individual ratings as input, it is clear that

natural noise is also present in the group recommendation process, which may bias results and diminish users satisfaction towards group recommendations.

This chapter aims at researching the natural noise management (NNM) in GRS to study its influence in group recommendation. A main difference with individual recommendation scenarios is that, from the group viewpoint, there are different levels of information in the ratings dataset. Therefore, existing NNM methodologies [239] are not directly applicable to group recommendation. Thus, it is necessary to propose NNM approaches specifically designed for GRS that consider the information levels present in GRSs.

A first step towards a NNM model for GRSs is to focus on members' ratings and reduce the natural noise with the aim of improving their quality, given that members' ratings are key data in aggregation-based GRSs [67] and their quality influences recommendation accuracy. This first approach can be formulated with the following hypothesis:

- **H1**: NNM using only the group ratings would improve the group recommendation.

A different approach to manage natural noise in GRSs is, given that NNM for individuals achieve clear improvements [239], to apply NNM models directly overlooking the group information. The rationale for this direct application is that GRSs approaches are supported by individual RSs [67], and it leads us to the following hypothesis:

- **H2**: NNM in the entire ratings database, disregarding the groups, would improve the group recommendation.

A further step in the management of natural noise for GRSs is to consider both the group and the global levels together. Such an approach would manage the natural noise at both levels tagging as noisy some ratings at the group level, but a different set of ratings at the global level. This behavior suggests that a hybrid

management could lead to better recommendation accuracy. This rationale is put forward in the following hypothesis:

- **H3**: managing natural noise in the entire ratings database and, after that, adding a second step that manages natural noise in the group ratings, would improve the results as compared to a single step of NNM.

Across this chapter, several approaches that apply these NNM strategies are presented to implement the presented hypotheses.

The remainder of this chapter is organized as follows. Section 6.2 presents four methods of natural noise management for group recommender systems. Section 6.3 presents the experiment done and its results. Finally, Section 6.4 summarizes this chapter.

6.2 Natural noise management in group recommendation

This section explores natural noise management in group recommendation. In the group recommendation scenario there are two levels of ratings: (i) *local level*, which is composed of the preferences that belong to group's members; and (ii) *global level*, which is composed of the preferences of all users in the recommender system dataset. Considering this feature of group recommendation together with the hypotheses posed in the introduction, four methods are presented to deal with NNM in both levels:

1. NNM-LL: Manage the natural noise in the local level using only local information, this is, apply NNM on group ratings using only themselves to detect inconsistencies.

2. NNM-LG: Manage the natural noise in the local level using global information. In this case, NNM is applied on group ratings, but all the ratings in the database are used to detect and correct inconsistencies.

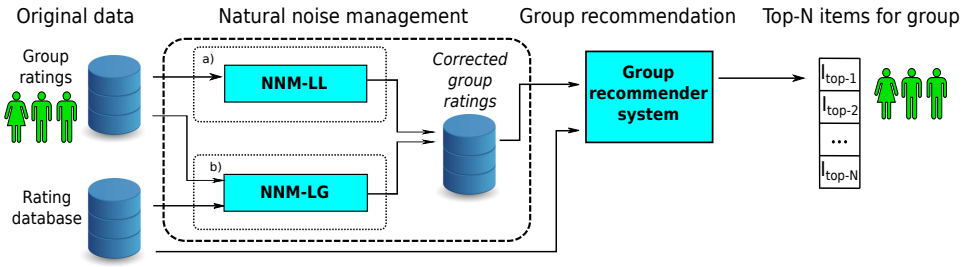


Figure 6.1: Scheme of local NNM for GRS, showing two approaches: a) NNM-LL, b) NNM-LG.

3. NNM-GG: Manage the natural noise in the global level using global information. This approach manages the noise in the entire dataset, but overlooking the composition of the group that will receive the recommendations.

4. NNM-H: Perform a cascade hybridization of previous approaches. This is, perform first a NNM-GG step to manage noise at global level, and apply NNM-LG over the results of the first approach in order to further refine the natural noise management considering the target group.

These four approaches are further detailed in the remaining of this section. In order to further clarify the way of functioning of the proposed approaches, Section 6.2.5 shows an illustrative example of application of the proposals.

6.2.1 Local natural noise management based on local information (NNM-LL)

Figure 6.1a) depicts the general scheme of the NNM-LL approach. Ratings of the target group G are first analyzed. After that, these ratings are corrected using only member's information. This proposal assumes that the information contained in the local level is enough to characterize natural noise, this is, that managing only the natural noise in the group ratings is enough to improve the dataset quality,

Algorithm 1 details the general way of computing group recommendations integrating natural noise management for groups based on group G_a members' ratings $R_{G_a,\bullet}$. Within Algorithm 1, Algorithm 2 implements the NNM process [239]. It

starts with the characterization of the user, item and rating itself. Notice that NNM-LL uses only local information R_{G_a,i_k} in the item classification (see Algorithm 2, line 4). This characterization is used to classify the ratings that do not follow the user or the item tendency as noisy. After that, noisy ratings are corrected with a new value predicted using collaborative filtering (see Algorithm 2, line 6)

Data: U,I,R,G
1 BuildRecommendationModel(U,I,R)
2 **foreach** G_a *in* G **do**
3 $\quad R^*_{G_a,\bullet}$ = NNMLL(G_a,$R_{G_a,\bullet}$)
4 $\quad recommendations_{G_a}$ = Recommend(G_a,$R^*_{G_a,\bullet}$)
5 **return** R^*

Algorithm 1: GRS with local NNM based on local information (NNM-LL).

Data: G_a,$R_{G_a,\bullet}$
Result: $R^*_{G_a,\bullet}$
1 **foreach** r_{m_l,i_k} *in* $R_{G_a,\bullet}$ **do**
2 $\quad c_{r_{m_l,i_k}}$ = Classify(r_{m_l,i_k})
3 $\quad c_{m_l}$ = Classify(m_l, $R_{m_l,\bullet}$)
4 $\quad c_{i_k}$ = Classify(i_k, R_{G_a,i_k})
5 $\quad$ **if** $(c_{m_l} = c_{i_k})$ *and* $(c_{m_l} \neq c_{r_{m_l,i_k}})$ *and* $(c_{m_l} \neq variable)$ **then**
6 $\quad\quad r^*_{m_l,i_k}$ = Predict(R,m_l,i_k)
7 $\quad$ **else**
8 $\quad\quad r^*_{m_l,i_k} = r_{m_l,i_k}$
9 **return** $R^*_{G_a,\bullet}$

Algorithm 2: Procedure for local NNM based on local information (NNM-LL)

The small amount of data used in NNM-LL, which is composed of only group G_a ratings, to perform this analysis and correction makes NNM-LL suitable to be applied when the recommendations are requested, which eliminates the need of storing a corrected dataset.

6.2.2 Local natural noise management based on global information (NNM-LG)

Figure 6.1b) specifies the NNM-LG approach. The way of detecting and correcting noisy ratings is similar to NNM-LL. However, the usage of only group ratings $R_{G_a i_k}$

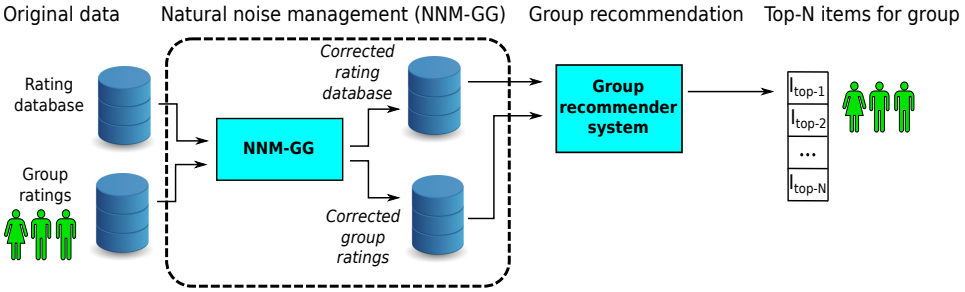


Figure 6.2: Global natural noise management based on global information (NNM-GG) application.

to characterize items might not be enough to properly classify items. Therefore, NNM-LG uses all the information available on the dataset regarding the item i_k for the characterization, this is, it uses all members ratings over the item $R_{\bullet i_k}$.

The same general scheme depicted in Algorithm 1 is followed to apply NNM-LG to a group recommender system. However, NNM-LG classifies items using all the information in the dataset $R_{\bullet,i_k}$, which modifies Algorithm 2, line 4 passing $R_{\bullet,i_k}$ instead of $R_{G_a i_k}$

The classification of items is done using more information in NNM-LG approach. This difference with NNM-LL approach is key in small groups because the usage of only a few ratings for items classification might lead to the assignment of a different class.

6.2.3 Global natural noise management (NNM-GG)

The general scheme for applying NNM-GG approach to a GRS is shown in Figure 6.2. NNM-GG applies NNM to the entire dataset before computing the recommendations, as stated in Algorithm 3. Similarly to a NNM that is applied to a RS for individuals [239], NNM-GG analyses all ratings in the database to detect and correct noisy ones. The specific procedure followed to manage noisy ratings is detailed in Algorithm 4.

The NNM-GG approach needs more computational resources, which makes it necessary to apply NNM-GG off-line. However, the off-line application of

Data: U,I,R,G
1 R^* = NNMGG(R)
2 BuildRecommendationModel(U,I,R^*)
3 **foreach** G_a *in* G **do**
4 $\quad$ $recommendations_{G_a}$ = Recommend(G_a, $R^*_{G_a}$)

Algorithm 3: GRS with global natural noise management.

Data: R
Result: R^*
1 **foreach** r_{u_j,i_k} *in* R **do**
2 $\quad$ $c_{r_{u_j,i_k}}$ = Classify(r_{u_j,i_k})
3 $\quad$ c_{u_j} = Classify(u_j, $R_{u_j,\bullet}$)
4 $\quad$ c_{i_k} = Classify(i_k, $R_{\bullet,i_k}$)
5 $\quad$ **if** $(c_{u_j} = c_{i_k})$ *and* $(c_{u_j} \neq c_{r_{u_j,i_k}})$ *and* $(c_{u_j} \neq variable)$ **then**
6 $\quad\quad$ $r^*_{u_j,i_k}$ = Predict(R,u_j,i_k)
7 $\quad$ **else**
8 $\quad\quad$ $r^*_{u_j,i_k} = r_{u_j,i_k}$
9 **return** R^*

Algorithm 4: Procedure for global natural noise management (NNM-GG)

NNM-GG allows to compute the recommendation model considering the corrected dataset R^*, i.e., the dataset without natural noise, which might offer better recommendation results. This improvement is possible because the influence of natural noise in the recommendation model is reduced.

6.2.4 Hybrid global-local natural noise management (NNM-H)

Figure 6.3 depicts the general scheme NNM-H application for group recommendation. NNM-H performs a cascade hybridization of NNM-GG and NNM-LG approaches. The cascade hybridization allows the results of the first NNM-GG step to be considered in the later NNM-LG step. Therefore, corrections made in the initial dataset by NNM-GG are considered when performing a group-wise natural noise management with NNM-LG.

The specific implementation of NNM-H is depicted in Algorithm 5. First, a global natural noise management is performed in line 1. After that, the corrected results are used to build the recommendation model. Later, a group-wise natural

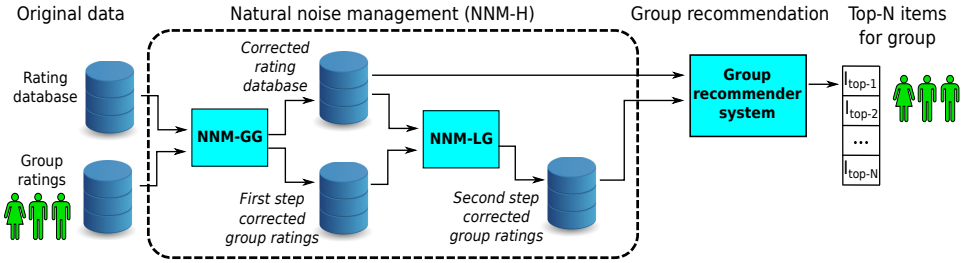


Figure 6.3: Hybrid global-local natural noise management (NNM-H) application to group recommendation.

noise management is performed in line 4. It is worth to remark that both the building of the recommendation model, and the group-wise natural noise correction use the corrected dataset R^*. This later step of NNM-LG generates a further refined dataset R^{**}, which is later used to calculate the group recommendation.

Data: U,I,R,G
1 R^* = NNMGG(R)
2 BuildRecommendationModel(U,I,R^*)
3 **foreach** G_a *in* G **do**
4 $\quad R^{**}_{G_a,\bullet}$ = NNMLG(G_a,$R^*_{G_a,\bullet}$)
5 $\quad recommendations_{G_a}$ = Recommend(G_a,$R^{**}_{G_a,\bullet}$)

Algorithm 5: GRS with hybrid natural noise management (NNM-H)

6.2.5 Illustrative example

In order to clarify the proposals, an illustrative example is presented in this section, which aims to clarify the integration of proposed NNM techniques with a GRS. Table 6.1 shows the initial dataset considered, where the target group G_a is composed of users u_1, u_2 and u_3,

In NNM-LL approach, only members' ratings $R_{G_a\bullet}$ are analyzed. Among all the rating analyzed, only $r_{u_2 i_2}$ is tagged as noise because its corresponding user u_1 is classified as *high*, its corresponding item i_2 is classified as high according to c_{i_k} using $R_{G_a i_k}$ and the rating itself is classified as *medium*. Both the user and item classes are different to those of the rating, therefore, the tendency of the rating cannot be supported with the evidence on either of them and rating $r_{u_2 i_2}$ is tagged

Table 6.1: Illustrative example for the classification of the ratings as possibly noisy.

	U							
	G_a						c_{i_k}	
	u_1	u_2	u_3	u_4	u_5	u_6	$R_{\bullet i_k}$	$R_{G_a i_k}$
i_1	5	5	5	5	5	2	high	high
i_2	5	3	5	3	3	3	medium	high
i_3	3	5	3	5	5	5	high	medium
i_4	5	5	5	5	5	5	high	high
i_5	1	1	4	2	1	5	low	low
c_{u_j}	high	high	high	high	high	high		

as possibly noisy.

In NNM-LG approach, members' ratings $R_{G_a\bullet}$ are analyzed but, instead of using only $R_{G_a i_k}$ for the item classification c_{i_k}, all ratings in the dataset $R_{\bullet i_k}$ are used for the classification. In this case, the ratings tagged as possibly noisy are r_{u_1,i_3}, and r_{u_3,i_3}.

In NNM-GG approach, all ratings in the dataset R are revised. Moreover, in item i_k classification c_{i_k} all the information is used, hence, the classification is done based on $R_{\bullet i_k}$. The ratings classified as possibly noisy by the NNM-GG approach are r_{u_1,i_3}, r_{u_3,i_3}, and r_{u_6,i_1}. Notice that this set of possibly noisy ratings is a superset of the ratings tagged as possibly noisy by NNM-LG.

In NNM-H approach, the ratings are first analyzed with NNM-GG to correct the possibly noisy ratings. Let assume that the corrected dataset R^* is the one shown in Table 6.2. The second step of the NNM-H approach is a cascade hybridization with NNM-LG, which uses R^* as input. Ratings in the group $R^*_{G_a\bullet}$ are analyzed using $R_{\bullet i_k}$ for classifying ratings. This process tags, as possibly noisy, the rating r_{u_3,i_3}.

6.3 Experimentation and evaluation

This section describes experiment done to measure the effect of proposed NNM approaches in the performance of GRSs. The remainder of this section is structured as follows. First, the configuration of techniques compared is described. After that,

Table 6.2: Corrected ratings R^* outputted by the NNM-GG correction applied over Table 6.1. The values corrected by NNM-GG are highlighted in bold. This rating database is R^* and it is used as input for NNM-LG to produce $R^{**}_{G_a,\bullet}$.

	U						
	G_a						c_{i_k}
	u_1	u_2	u_3	u_4	u_5	u_6	$R_{\bullet i_k}$
i_1	5	5	5	5	5	**3**	high
i_2	5	3	5	3	3	3	medium
i_3	**4**	5	**3**	5	5	5	high
i_4	5	5	5	5	5	5	high
i_5	1	1	4	2	1	5	low
c_{u_j}	high	high	high	high	high	high	

the datasets and methods for processing them are detailed. Later, the evaluation measures are defined. Eventually, the results of the techniques compared are shown and analyzed. For the sake of clearness, the results are shown separately for each GRS aggregation approach: (i) recommendation aggregation-based GRSs and, (ii) rating aggregation-based GRSs. Finally, a discussion of the results is done in order to test the hypotheses posed in the introduction of this chapter.

6.3.1 Techniques compared

In section 6.1, three hypotheses are posed, which led us to the proposal of four NNM approaches. All these four approaches are evaluated and compared against a baseline approach, which does not perform any natural noise management. Therefore, five techniques are compared in the experiment: (i) Baseline, (ii) NNM-LL, (iii) NNM-LG, (iv) NNM-GG, and (v) NNM-H.

All these techniques focus on managing the natural noise in a dataset for a GRS to recommend based on it. Therefore, the effect of each of these approaches must be evaluated for various GRSs in order to thoroughly study their performance. Specifically, this experiment studies NNM performance regarding rating aggregation GRSs and recommendation aggregation GRSs [67], given that each of them manages users' preferences in a different way. The effects of the NNM proposals on each GRS approach are analyzed separately.

Moreover, each of these aggregation approaches can be carried out applying a different aggregation opperator. The techniques are compared for the aggregation operators Mean and Min, given that several works [17, 73] have pointed out that they achieve the best results in GRSs.

These GRSs use an individual recommender system, which is also a variable that we consider in the experiment. The NNM approaches are checked for GRSs based on both item-based and user-based collaborative filtering. They use the Pearson's correlation coefficient as similarity measure to find the 100 best neighbors, apply a relevance factor of 20 [41], and use the adjusted weighted sum as rating predictor [190].

6.3.2 Data set

The datasets used in this case study are:

- The MovieLens 100k dataset. It was collected by GroupLens Research Project at the University of Minnesota (http://grouplens.org). The MovieLens 100k dataset is composed of 100,000 ratings given by 943 users over 1,682 movies in the one to five stars domain.

- The Netflix Tiny dataset. It is a small version of Netflix dataset provided with the Personalized Recommendation Algorithms Toolkit (http://prea.gatech.edu). The Netflix Tiny dataset is composed of 4,427 users, 1,000 items, and 56,136 ratings, which are also given in the one to five stars domain. This dataset has a high sparsity, which may bias the results. Therefore, only users with 10 or more ratings are used, which remains 1,757 users in the dataset used for the experimentation.

None of those dataset contains information about groups. Users are grouped randomly in order to evaluate the results for occasional groups [67, 228], specifically, with group sizes of 5, 10 and 15. Larger group sizes were excluded from the experiment because they are not used in these kinds of experimental scenarios [67, 73].

The hold-out validation is used to split the dataset into training and tests sets and it is applied with a 20% test set. This validation has been executed 20 times and the evaluation measures results shown are the average values.

6.3.3 Experiment results for recommendation aggregation-based GRS

The results for the evaluated NNM approaches on GRS based on recommendation aggregation are presented in this section. Table 6.3 shows the recommendation results for MovieLens 100k and Netflix Tiny datasets. In the first three columns the configuration of the GRS is shown, starting with dataset. The second column specifies the single-user RSs IB and UB and the aggregation approaches Avg and Min, which are shown together as the prediction technique. The third column specifies the group size. In order to make the results more readable, the best result for each row is highlighted using bold font, which means that it is the best result across NNM techniques.

The results clearly show that, in general, the application of NNM approaches make a group recommender system to achieve better recommendations, which suggests that the techniques compared successfully reduce the natural noise in the dataset. The magnitude of the improvement is different for each NNM approach, hence, we analyze them separately.

In the case of NNM-LL, its application does not implies a significant improvement in performance, as its results in all cases show. Compared to the baseline, its performance does not improves significantly, which suggests that the characterization of items done in it does not lead to reducing the natural noise effectively. Therefore, the application of NNM-LL solely is not enough to manage the natural noise.

In the case of NNM-LG, its application results in a slight improvement of the recommendation quality as compared to the results of the baseline method. If we focus on each GRS, the magnitude of these improvements are different. Specifically, IB+Min results improve by 0.01, while for UB+Avg the improvement in rec-

Table 6.3: Results of mean absolute error for the compared natural noise management approaches on recommendation aggregation group recommender systems.

Dataset	Prediction Technique	Group size	Base	NNM-LL	NNM-LG	NNM-GG	NNM-H
MovieLens 100k	IB+Avg	5	0.8779	0.8778	0.8747	0.8607	**0.8583**
		10	0.8998	0.8996	0.8956	0.8837	**0.8804**
		15	0.9080	0.9079	0.9036	0.8913	**0.8875**
	IB+Min	5	1.0218	1.0214	1.0137	1.0045	**0.9983**
		10	1.1404	1.1403	1.1328	1.1263	**1.1200**
		15	1.2066	1.2066	1.1959	1.1918	**1.1830**
	UB+Avg	5	0.8053	0.8053	0.8051	0.7853	**0.7852**
		10	0.8127	0.8127	0.8125	0.7932	**0.7931**
		15	0.8146	0.8145	0.8145	**0.7946**	**0.7946**
	UB+Min	5	0.8421	0.8419	0.8399	0.8190	**0.8172**
		10	0.8700	0.8698	0.8674	0.8463	**0.8444**
		15	0.8845	0.8844	0.8815	0.8593	**0.8572**
Netflix Tiny	IB+Avg	5	0.8435	0.8431	0.8415	0.8386	**0.8368**
		10	0.8630	0.8627	0.8598	0.8572	**0.8542**
		15	0.8627	0.8626	0.8595	0.8569	**0.8539**
	IB+Min	5	1.0074	1.0072	1.0025	1.0011	**0.9963**
		10	1.1451	1.1445	1.1330	1.1364	**1.1262**
		15	1.2252	1.2251	1.2117	1.2169	**1.2046**
	UB+Avg	5	0.8127	0.8128	0.8130	**0.8060**	0.8062
		10	0.8245	0.8245	0.8244	0.8174	**0.8173**
		15	0.8194	0.8194	0.8193	**0.8129**	**0.8129**
	UB+Min	5	0.8616	0.8615	0.8609	0.8538	**0.8532**
		10	0.9034	0.9033	0.9020	0.8945	**0.8934**
		15	0.9192	0.9192	0.9187	0.9108	**0.9103**

ommendation quality is not significant because its is magnitude is less than 0.001. In the remaining cases, IB+Avg and UB+Min, the improvement of NNM-LG is narrow. Compared to NNM-LL, the results of NNM-LG are better, which suggests that the usage of more information to characterize items benefits the natural noise identification within recommendation aggregation-based GRSs.

In the case of NNM-GG, its application makes the recommendation aggregation-based GRS achieve better results as compared to the baseline, NNM-LL and NNM-LG. The application of NNM-GG was expected to achieve better improvements for a GRS because previous approaches for single user RSs achieved a similar improvement of a recommender system [239]. Moreover, the impact of NNM-GG on a GRS was expected to be of greater magnitude because it analyses more ratings to

reduce natural noise.

In the case of NNM-H, its application achieved the best results for the evaluated cases regarding recommendation aggregation-based GRSs, which is an evidence that the hybridization of NNM-GG and NNM-LG outperforms their performance applied solely for the natural noise management. NNM-H was the best NNM approach for IB+Avg, IB-Min and UB+Min prediction techniques.

6.3.4 Results in rating aggregation-based GRS

The results for the evaluated NNM approaches on GRS based on rating aggregation are presented in this section. Table 6.4 shows the recommendation results for MovieLens 100k and Netflix Tiny datasets, which are presented in a similar way to those of recommendation aggregation-based GRSs.

The results show, in general, a similar performance to those of the NNM approaches applied to recommendation aggregation-based GRSs. Also, the magnitude of the improvements is different for each NNM approach, therefore, each of them is analyzed separately.

In the case of NNM-LL, the results achieve a performance similar to those of the baseline, which suggests that the use of local information is not enough to manage the natural noise for the improvement of a GRS.

In the case of NNM-LG, the results show that GRSs achieved a slight improvement as compared to the baseline. It provided improvements in IB+Min prediction approach for larger groups on both datasets. The reason for the larger magnitude of the improvement on those cases might be due to the larger error obtained by the baseline in IB+Min as compared to IB+Avg, which means that there is a larger margin for improvement and NNM-LG is able to take advantage of it.

In the case of NNM-GG, the results show that it improved the accuracy of a GRS as compared to the performance of the baseline, NNM-LL and NNM-LG. This improvement might be rooted on the larger amount of ratings corrected by NNM-GG, which suggests that it is more effective in terms of natural noise reduc-

Table 6.4: Results of mean absolute error for the compared natural noise management approaches on rating aggregation group recommender systems.

Dataset	Prediction Technique	Group size	Base	NNM-LL	NNM-LG	NNM-GG	NNM-H
MovieLens 100k	IB+Avg	5	0.8664	0.8664	0.8657	0.8477	**0.8473**
		10	0.8898	0.8898	0.8884	0.8723	**0.8712**
		15	0.8990	0.8990	0.8969	0.8814	**0.8797**
	IB+Min	5	0.8877	0.8874	0.8808	0.8674	**0.8617**
		10	0.9563	0.9559	0.9387	0.9347	**0.9197**
		15	1.0252	1.0245	0.9942	1.0006	**0.9770**
	UB+Avg	5	0.7998	0.7997	0.7997	**0.7801**	0.7802
		10	0.8101	0.8101	0.8102	0.7913	**0.7912**
		15	0.8133	0.8133	0.8135	**0.7937**	0.7938
	UB+Min	5	0.8019	0.8017	0.8013	0.7820	**0.7818**
		10	0.8139	0.8138	0.8136	0.7944	**0.7943**
		15	0.8188	0.8188	0.8186	**0.7983**	**0.7983**
Netflix Tiny	IB+Avg	5	0.8382	0.8381	0.8376	0.8330	**0.8325**
		10	0.8637	0.8638	0.8624	0.8579	**0.8566**
		15	0.8682	0.8682	0.8663	0.8621	**0.8604**
	IB+Min	5	0.8689	0.8683	0.8643	0.8628	**0.8585**
		10	0.9351	0.9344	0.9234	0.9282	**0.9163**
		15	0.9809	0.9801	0.9630	0.9738	**0.9564**
	UB+Avg	5	0.8092	0.8094	0.8092	**0.8025**	0.8027
		10	0.8221	0.8223	0.8222	**0.8152**	0.8153
		15	0.8178	0.8178	0.8178	0.8111	**0.8110**
	UB+Min	5	0.8143	0.8143	0.8134	0.8075	**0.8067**
		10	0.8267	0.8269	0.8265	0.8203	**0.8198**
		15	0.8226	0.8226	0.8223	0.8162	**0.8158**

tion.

In the case of NNM-H, the results obtained were, overall, the best ones as compared to other techniques applied to rating aggregation-based GRSs.

6.3.5 Discussion

This section aims to test the hypotheses posed in Section 6.1 analyzing the results of the experimentation. The results obtained determine that **H1 is rejected**, hence the management of natural noise only in group ratings is not enough to reduce the natural noise and allow a GRS to achieve better results. Regarding the second hypothesis, **H2 is accepted**, hence, disregarding group information in the natural noise management and manage noise in the entire database improve the results

Table 6.5: Paired samples t-test p-values to compare each natural noise management technique with the baseline on MovieLens 100k dataset.

Dataset	Group aggregation	Prediction technique	Group size	NNM-LL	NNM-LG	NNM-GG	NNM-H
MovieLens 100k	Rating	IB+Avg	5	0.1573	**<0.0001**	**<0.0001**	**<0.0001**
			10	0.1770	**<0.0001**	**<0.0001**	**<0.0001**
			15	0.6921	**<0.0001**	**<0.0001**	**<0.0001**
		IB+Min	5	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
			10	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
			15	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
		UB+Avg	5	0.9587	0.9225	**<0.0001**	**<0.0001**
			10	**0.0245**	0.2663	**<0.0001**	**<0.0001**
			15	0.1083	**0.0013**	**<0.0001**	**<0.0001**
		UB+Min	5	0.3497	0.0774	**<0.0001**	**<0.0001**
			10	0.1861	0.2058	**<0.0001**	**<0.0001**
			15	0.7174	0.0922	**<0.0001**	**<0.0001**
	Recommendation	IB+Avg	5	**0.0134**	**<0.0001**	**<0.0001**	**<0.0001**
			10	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
			15	**0.0005**	**<0.0001**	**<0.0001**	**<0.0001**
		IB+Min	5	**0.0005**	**<0.0001**	**<0.0001**	**<0.0001**
			10	0.2681	**<0.0001**	**<0.0001**	**<0.0001**
			15	0.1190	**<0.0001**	**<0.0001**	**<0.0001**
		UB+Avg	5	0.6047	0.1038	**<0.0001**	**<0.0001**
			10	0.9303	**0.0304**	**<0.0001**	**<0.0001**
			15	0.7342	**0.0096**	**<0.0001**	**<0.0001**
		UB+Min	5	**0.0275**	**<0.0001**	**<0.0001**	**<0.0001**
			10	**0.0003**	**<0.0001**	**<0.0001**	**<0.0001**
			15	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**

of a GRS. Hypothesis **H3 is accepted**, therefore, the hybridization of a first step of natural noise management in the entire database with a second step of natural noise management focused on the target group improves the results as compared to the application of each of these steps solely. The remainder of this section further details how we determined the acceptance and rejection of those hypotheses based on the evidence gathered in the experiment.

H1: Managing natural noise in the group ratings only would improve the GRS.

The proposal of local level approaches NNM-LL and NNM-LG is done to implement the ideas posed in hypothesis H1. Therefore, in order to test it, the results of those approaches are compared to those of the baseline. In general, nar-

row improvements are achieved by local level approaches. In the case of rating aggregation-based GRSs with IB+Min, NNM-LG overcomes the results of the baseline. The results are tested to determine whether they are statistically significant apply the paired samples t-test. This test analyses the differences obtained by two methods to determine if the difference between them might appear due to random chance, which makes the results not statistically significant, or the difference is due to the methods being different enough. Specifically, the paired t-test is applied to the compare the results of each NNM technique with those of the baseline.

Tables 6.5 and 6.6 show the p-values for each of the tested cases in MovieLens and Netflix Tiny datasets, respectively. The tests that were able to reject the equality with a confidence level of 95% have been highlighted.

The p-values on NNM-LL and NNM-LG columns determine, mostly, that the results are not statistically significant, although for a limited number of cases the results are statistically significant. Specifically, NNM-LL results improve for both datasets in rating aggregation-based GRSs with IB+Min, and for recommendation aggregation with IB+Avg. In the case of NNM-LG, the differences with the baseline are statistically significant for IB+Avg and IB+Min in both datasets.

Given that there are statistically significant differences only in a limited number of cases, hypothesis **H1 is rejected**. We can conclude that the application of local-based techniques does not provide significant improvements to a group recommender system.

H2: Managing natural noise in the entire ratings database, disregarding the groups, would improve the group recommendation.

The hypothesis H2 is tested analyzing the results of NNM-GG as compared to the results of the baseline. In general, Table 6.3 and 6.4 show that NNM-GG improves the results of the baseline regarding the mean absolute error with an average magnitude of 0.02. Similarly to H1, the results are tested to establish whether they are statistically significant. The p-values are shown in column NNM-GG both in Table

Table 6.6: Paired samples t-test p-values to compare each natural noise management technique with the baseline on Netflix Tiny dataset.

Dataset	Group aggregation	Prediction technique	Group size	NNM-LL	NNM-LG	NNM-GG	NNM-H
Netflix Tiny	Rating	IB+Avg	5	0.3459	**0.0104**	<**0.0001**	<**0.0001**
			10	0.3524	<**0.0001**	<**0.0001**	<**0.0001**
			15	0.1991	<**0.0001**	<**0.0001**	<**0.0001**
		IB+Min	5	**0.0040**	<**0.0001**	<**0.0001**	<**0.0001**
			10	<**0.0001**	<**0.0001**	<**0.0001**	<**0.0001**
			15	<**0.0001**	<**0.0001**	<**0.0001**	<**0.0001**
		UB+Avg	5	0.1121	0.9215	<**0.0001**	<**0.0001**
			10	0.1968	0.5745	<**0.0001**	<**0.0001**
			15	0.6876	0.9641	<**0.0001**	<**0.0001**
		UB+Min	5	0.6833	0.1541	<**0.0001**	<**0.0001**
			10	**0.0154**	0.6594	<**0.0001**	<**0.0001**
			15	0.7768	0.3047	<**0.0001**	<**0.0001**
	Recommendation	IB+Avg	5	**0.0112**	<**0.0001**	<**0.0001**	<**0.0001**
			10	**0.0002**	<**0.0001**	<**0.0001**	<**0.0001**
			15	**0.0002**	<**0.0001**	<**0.0001**	<**0.0001**
		IB+Min	5	0.2777	<**0.0001**	<**0.0001**	<**0.0001**
			10	0.0886	<**0.0001**	<**0.0001**	<**0.0001**
			15	0.3985	<**0.0001**	<**0.0001**	<**0.0001**
		UB+Avg	5	0.0635	0.1132	<**0.0001**	<**0.0001**
			10	0.9263	0.4457	<**0.0001**	<**0.0001**
			15	0.3817	0.3929	<**0.0001**	<**0.0001**
		UB+Min	5	0.8274	0.0894	<**0.0001**	<**0.0001**
			10	0.5147	**0.0033**	<**0.0001**	<**0.0001**
			15	0.4580	0.0997	<**0.0001**	<**0.0001**

6.5 and 6.6, where bold p-values denote statistically significant results. All tests are able to reject the equality of results for NNM-GG, therefore, the evidence suggests that global NNM approach is able to improve the results of a group recommender system across the datasets, aggregation approaches and aggregation operators considered in the experiment.

Therefore, the hypothesis **H2 is accepted**. The improvement achieved by NNM-GG as compared to the baseline might be caused by the correction of the entire dataset used for building the recommendation model of the single user RS that the group recommender system uses. This fact results in better predictions, which allows a GRSs to produce better recommendations.

Table 6.7: Paired samples t-test p-values to compare NNM-GG with NNM-H.

		Group aggregation			
		Rating		Recommendation	
Prediction technique	Group size	MovieLens 100k	Netflix Tiny	MovieLens 100k	Netflix Tiny
IB+Avg	5	**0.0013**	**0.0136**	**<0.0001**	**<0.0001**
	10	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
	15	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
IB+Min	5	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
	10	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
	15	**<0.0001**	**<0.0001**	**<0.0001**	**<0.0001**
UB+Avg	5	0.2662	0.5664	0.1273	0.4065
	10	0.3890	0.6813	0.1118	0.6998
	15	**0.0466**	0.6162	0.5110	0.4616
UB+Min	5	0.3643	0.2169	**<0.0001**	0.3520
	10	0.5637	0.3818	**<0.0001**	**0.0040**
	15	0.7230	0.1835	**<0.0001**	**0.0250**

H3: Managing natural noise in the entire ratings database and, after that, adding a second step that manages natural noise in the group ratings, would improve the results as compared to a single step of NNM.

The hypothesis H3 can be tested comparing the results of NNM-GG with those achieved by NNM-H. NNM-GG results are, in general, overcome by those of the NNM-H approach. Similarly to the test of previous hypotheses, the results are tested using the paired samples t-test to compare NNM-GG and NNM-H on the various configurations explored in the experiment. The p-values are depicted in Table 6.7, which suggests that in general the results obtained are statistically significant. Specifically, NNM-H approach achieves statistically significant results in all configurations with IB. Regarding UB, NNM-H obtains statistically significant results for recommendation aggregation with UB+Min.

Therefore, **H3 is accepted**, for IB prediction techniques and UB+min with recommendation aggregation. In the remaining cases there is no evidence of differences between NNM-GG and NNM-H. We can conclude that the best approach for managing natural noise in group recommendation with IB prediction technique is the NNM-H approach. It is also the best one for recommendation aggregation-

based GRSs based on UB that use the minimum as aggregation operator.

6.3.6 Complexity and deployment

The analysis of the results concluded that NNM-H is the best technique for GRSs based on IB. Therefore, this section focuses on the analysis of the deployment of such a GRS. Here, we consider two important aspects of a real-world GRS: (i) the complexity order of the NNM-H approach, and (ii) the frequency of update of the recommendation model. Therefore, this section aims to study the computational complexity of NNM-H approach. Moreover, some suggestions are provided regarding the deployment of NNM-H approach on GRSs with IB prediction.

1. *NNM-H computational complexity*

   NNM-H approach applies first a NNM-GG step before the IB recommendation model is built. After that and before the group recommendation is requested, a NNM-LG step is applied to the group ratings. Therefore, the complexity order of each phase of the NNM-GG approach is studied separately. First, we analyze NNM-GG, which is composed of two phases:

   - Rating detection: all ratings in the dataset are evaluated to detect the noisy ones. The rating detection complexity order is $O(|U| \cdot |I|)$.
   - Rating correction: the value of each rating tagged as possibly noisy is replaced with a corrected value that is computed using UKNN. The noisy ratings of each user can be grouped to be corrected all at once (optimization over Algorithm 4) allowing to compute the neighborhood only once per user, which is the costly part of the correction phase with a computational complexity of $O(|U| \cdot |I|)$. The computational complexity of the rating correction for all users is $O(|U|^2 \cdot |I|)$.

   Previous research [159] stated that $|U| \gg |I|$. Therefore, the complexity order of NNM-GG can be safely expressed as $O(|U|^2)$. The rating detection

and rating correction are executed one after the other. Given that the computational complexity of the rating correction dominates the complexity of the rating correction, the computational complexity of NNM-GG is $O(|U|^2)$.

Secondly, NNM-LG is analyzed, which is also composed of two phases:

- Rating detection: the ratings in the group $R_{G_a\bullet}$ are evaluated to tag noisy ones, therefore, the rating detection complexity order is $O(|G_a| \cdot p)$, where $p = max(|R_{m_k,\bullet}|)$ and $m_k \in G$.
- Rating correction: the correction is done similarly to NNM-GG, but instead of correcting ratings in the whole dataset, only ratings within group G_a are corrected. Therefore, the computational complexity of this phase is $O(|U| \cdot |G_a|)$

Here, the complexity of the rating correction dominates those of the rating detection because p value is restricted by $|I|$ and $|U| \gg |I|$ [159]. Summarizing, the complexity added to the model computation for NNM-H is $O(|U|^2)$, and the complexity added to the recommendation computation is $O(|U| \cdot |G_a|)$.

2. *Deployment of a GRS with NNM-H and IB prediction*

GRSs based on IB prediction generate a recommendation model, whose calculus have a computational complexity order $O(|I|^2 \cdot |U|)$ [159]. This recommendation model can be computed offline in a deployed GRS, and it is often updated with certain frequency, such as daily or weekly, to consider new information [242]. The integration of NNM-H is straightforward, and it is done executing first the NNM-GG step before generating the recommendation model.

However, when the number of items grows it is not affordable to perform a complete model updated. This is particularly important in domains with a large number of items or with a high data variation, such as personalized

advertising or news recommendation. The size of the dataset in such a domain would result in the requirement of more computational resources for the model update. Incremental models were proposed to reduce the resources needed in these situations [151], where there are partial updates when new ratings are added to the dataset. When a new rating r_{ui} is added, a noise detection and correction is performed focuses on it. In the case of a change in the item or the user classification [239], it is needed to trigger a NNM process over the corresponding user or item ratings. These tasks require low computational cost as compared to the cost of updating the model incrementally.

In conclusion, the low computational resources needed by NNM-H approach are out-weighted by the benefits that it provides in terms of recommendation accuracy.

6.4 Summary

This chapter details four approaches aimed to manage the natural noise to allow a group recommender system achieve better recommendation results. The four approaches NNM-LL, NNM-LG, NNM-GG, and NNM-H follow the thoughts stated in three hypotheses posed in section 6.1. The results of the experimentation carried out show that the application of NNM benefits group recommendation. Specifically, the hybridization of global and local noise management provided the best results overall. The experimentation is completed with a computational complexity analysis. The evidence gathered in the experiment together with the computational complexity analysis suggests that the application of NNM-H for group recommendation provides accuracy improvements that out-weights the need for additional computations. Therefore, the application of NNM-H is beneficial for most GRSs, and it provided greater benefits when it is applied to GRSs that use item-based collaborative filtering.

Chapter 7

Context-aware question answering recommendation with semantic model

7.1 Introduction

Previous chapters propose several recommendation models to deliver successful recommendations targeted to groups of users. Among the proposed approaches, there are proposals that focus on keeping the group information for the recommendation, that apply consensus processes to group recommendation, that apply opinion dynamics models and that manage noise in group ratings. These proposals pose an advance in group recommendation. However, researchers pointed out context-aware recommendations as an interesting research trend within recommendation.

In this chapter, we focus on the extension of recommender systems to provide context-aware recommendations [8, 127]. An interesting source of information are microblogging services. This information can be used by a recommender system in order to analyze and understand collaborative trend interest as the current context. Therefore, our proposal uses status updates from microblogging services, such as Twitter, as the source of current trend interest. Formally, a status update consists of a free text input generated by a user with certain timestamp, among other metadata.

The community-based question answering domain produces large amounts of data, similarly to microblogging services. Hence, recommender systems are widely applied. In this chapter, we focus on the recommendation of question answering (QA) items with content-based approach and contextual information integration [127]. In the community-based question answering domain, users constantly submit new questions for other users to answer them. Content-based recommendation is particularly useful to cope with item cold-start, which is produced by the addition of items to the system. Specifically, among the approaches for content-based recommendation, content-based approach with textual description is suitable to be applied in the question answering domain because QA items have a strong component of textual information for both formulating the question and answering it [79].

Several approaches were proposed to deal with question answering recommendation with content-based approaches. Shao et al.[208] propose to label questions in a latent semantic feature category using Latent Dirichlet Allocation, which is later used to find the most suitable answerers, i.e., the best users that can answer a given question. A similar approach was proposed by Zheng et al. [252], which apply trust-based analysis of answerers together with content analysis. Those two approaches search for the best answerer for new questions in order to reduce the answer time. Instead, this chapter proposal focuses on the recommendation of already answered questions to expand users' knowledge. In this direction, Odiete et al. [163] build a graph of expertise from users' preferences and suggest those questions that cover the concepts of the gaps of knowledge found in the graph.

An interesting question answering task is to recommend already answered questions that are suitable for the target user's area of interest and are relevant regarding current collaborative trend interest. Such systems are known as context-aware content-based recommender systems [66]. DePessemier et al. [66] provide context-awareness to recommendation in mobile devices through the data provided by their sensors to recommend news to users. Other examples of context-aware

content-based recommendation, such as SeaHawk [182] and Prompter [181] use the location of the source code from which the recommendation is requested as contextual information to improve the recommendation of StackOverflow questions. Libra [183] has a similar approach but it also includes resources previously checked by the user to complete its knowledge about the user context. In e-commerce scenarios, Parikh et al. [174] use current buzzwords as an indication of contextual interest to present products that satisfy current trend. No previous research proposed a model for recommendation in question answering domain considering current trend interest.

A model for context-aware content-based recommendation for the question answering domain is introduced in this research. The contextual information of this model is extracted from microblogging systems, which provides the system with the immediacy they have [105]. This information is used to characterize current trends in the collaborative trend interest, which helps at recommending answered question that are related with it. This feature indirectly reduces the over-fitting problem of content-based approaches. The model clusters the context in order to identify topics, which are later selected regarding the target user's interests. These selected topics influence the user model to integrate context in the profiles, which enables context-aware recommendation.

The domains considered in such a model are characterized by the large-scale data in the case of question answering, and the generation of data at high rate in the case of microblogging services. In order to cope with the scalability requirements of such a system, the proposal is developed within Spark, a distributed *big data framework* that takes advantage of in-memory operations.

The remainder of this chapter is structured as follows. The proposal for context-aware content-based recommendation in the question answering domain is introduced in Section 7.2. Section 7.3 presents the experimentation and evaluation done to validate the suitability of the model. Finally, Section 7.4 provides a summary of this chapter.

7.2 Context-aware content-based recommender system for questions based on topic detection in current trend interest

This section proposes LSAContextCluster, a context-aware content-based recommender system. This model is applied in situations in which the recommendation of answered question can be influenced by specific context. For example, consider a system that recommends question in the history domain. If currently people are posting status updates about Columbus Day, the system might promote questions about the discovery of the Americas. With the aim of providing this behavior, the proposal includes contextual information in users' profiles before computing the recommendation. The recommender system would then compute the recommendations, which are adjusted to both the user's preferences and the current context.

Figure 7.1 shows the general scheme of LSAContextCluster. It follows the contextual modeling scheme for integration of contextual information, given that it considers context when building the recommendation model. LSAContextCluster is composed of five phases:

(i) QA domain semantic analysis: The LSA modeling is applied to analyze the term-document matrix and reduce its dimensionality.

(ii) Build user's preference profile: A preference profile is generated for each user aggregating the semantic profile of those items that the user liked in the past.

(iii) Build context model: The stream of recent status updates is analyzed applying clustering to separate them in several topics. A context topic profile is calculated for each topic in the LSA space.

(iv) Contextualize user's profiles: The most similar context to the target user's preferences is selected and their profile combined to produce the contextualized user's profile.

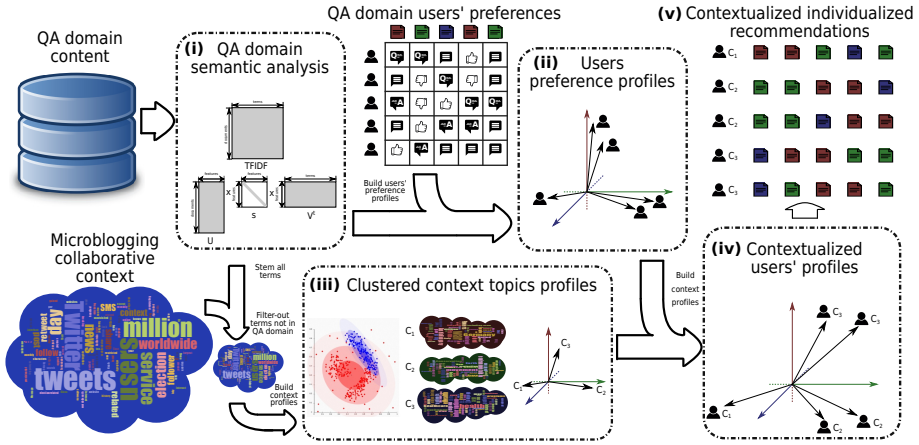


Figure 7.1: General scheme of the proposed context-aware content-based recommender system for questions based on topic detection in current trend interest.

(v) Prediction: The contextualized user's profile is compared with documents' profiles and the most suitable documents are recommended.

7.2.1 QA domain semantic analysis

The first phase of LSAContextCluster is the QA domain semantic analysis. The community-based question answering dataset is assumed to contain textual information of the questions and answers, in which several answers can be associated to one question. The text of a given question and its answers are combined together, and the resulting text is processed. The Porter Stemmed algorithm [184] is applied to stem the terms. After that, the TFIDF approach is applied to build the term-document matrix.

LSA is applied to the term-document matrix to reduce its dimensionality. This is done decomposing the term-document matrix in a matrix A that represents all terms features, a vector s with the singular values, and matrix B with all items features (see Eq. 2.6). This decomposition is done with singular value decomposition, which reduces original matrix dimensionality and keeps its f most relevant singular values. A and B contains the profiles of terms (see Eq. 7.1) and items (see Eq.

7.2), respectively, in the feature space.

$$profile_{t_j}^{LSA} = \{a_{t_j,1}, \ldots, a_{t_j,f}\} \tag{7.1}$$

$$profile_{i_k}^{LSA} = \{b_{i_k,1}, \ldots, b_{i_k,f}\} \tag{7.2}$$

7.2.2 Building user's preference profile

In this phase, the user's profile is built upon the profiles of the items that are relevant regarding user's preferences. Users' profiles need to be expressed in the same feature space.

The system holds a unary matrix that states whether a given user has expressed interest in a given item, either creating, commenting or voting it. The set I_{u_j} contains the items that are relevant for user u_j. The profile of a user u_j is given as:

$$profile_{u_j}^{LSA} = \sum_{i_k \in I_{u_j}} profile_{i_k}^{LSA} = \{ \sum_{i_k \in I_{u_j}} profile_{i_k,1}^{LSA}, \ldots, \sum_{i_k \in I_{u_j}} profile_{i_k,f}^{LSA} \} \tag{7.3}$$

This user's profile $profile_{u_j}^{LSA}$ is not normalized because the cosine correlation coefficient is used to compare profiles, which considers the angle of the vectors compared and disregards their magnitude.

7.2.3 Context model building

In this phase, the context model is built using the term-document description in the reduced feature space and the information gathered from the microblogging service.

In this model, items that are relevant to the current trend interest are promoted in the recommendation. Current trend interest, i.e., the context, is extracted from the status updates gathered from microblogging services, such as Twitter. These status updates contain a free text input that users post, together with meta-data such as the time-stamp. These status updates are used to build the context model. Figure

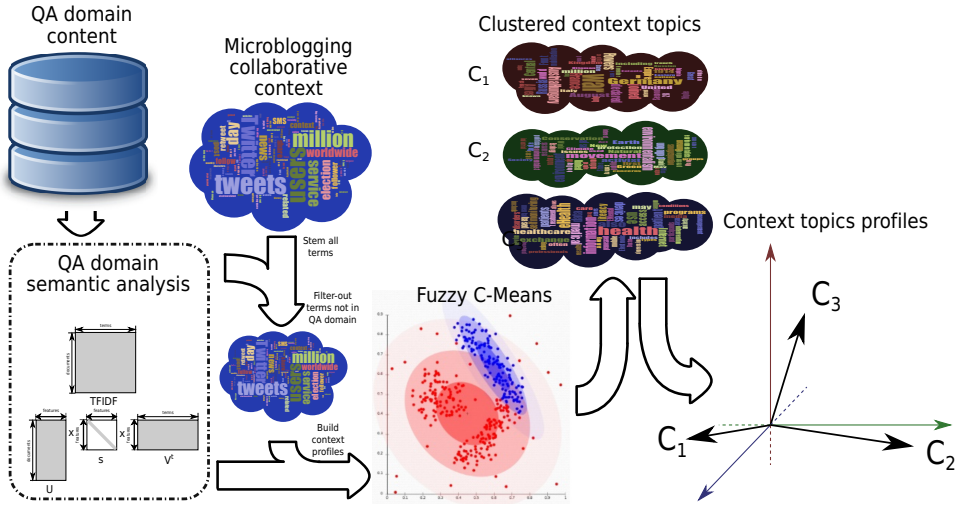


Figure 7.2: Context model building phase.

7.2 shows the scheme of the context model building phase.

The status updates gathered in a given time window compose the context. This time window is set to 24 hours in this model, although other configurations might be used in different scenarios. The contextual model phase begins stemming all words in the status updates. After that, LSAContextCluster filters the terms that do not appear in the semantic model (see Section 7.2.1).

Each of the status update might be related to a topic. Therefore, the identification of the topics present in the context is a key task for a suitable contextual modeling. In order to identify such topics, the proposal applies fuzzy c-means clustering algorithm [33]. For such task, the term representation used is their profile $profile_{t_j}^{LSA}$, and the cosine correlation coefficient is used as profile distance. As a result, the clustering method outputs a set of clusters. Each of these clusters, noted as c_l, is a topic.

Afterwards, a context profile is generated for each topic identified. This profile is generated considering the profiles of the terms that compose the target cluster c_l

as shown in Eq. 7.4. These topic profiles compose the context model.

$$profile_{c_i}^{LSA} = \sum_{t \in c_i} profile_t^{LSA} = \{\sum_{t \in c_i} profile_{t,1}^{LSA}, \ldots, \sum_{t \in c_i} profile_{t,f}^{LSA}\} \tag{7.4}$$

7.2.4 Users' profile contextualization

In this phase, users' profiles are contextualized using the context topic profiles generated in previous phase to provide contextualized recommendations that are relevant regarding user's preferences. To do so, each context profile $profile_{c_l}^{LSA}$ is analysed and the most similar one to user's profile $profile_{u_k}^{LSA}$ is selected using the cosine correlation coefficient.

$$\underset{c_l}{\text{argmax}} \quad cosine(profile_{u_k}^{LSA}, profile_{c_l}^{LSA}) \tag{7.5}$$

This way, the user's profile is contextualized using the context topic most similar to the user's preferences, and a personalized contextualization process can be carried out. The contextualized user's profile is generated combining the selected context topic profile and the user preference profile. This combination is done with the convex combination.

$$profile_{C,u_k}^{LSA} = \alpha * profile_{u_k}^{LSA} + (1-\alpha) * profile_{c_l}^{LSA} \tag{7.6}$$

where parameter α regulates the importance of the user preference profile over the context topic profile.

7.2.5 Prediction

In this phase, the suitability of a given item i_k regarding to the contextualized user's profile is computed according to the following equation:

$$p_{u_j,i_k,C} = profile_{C,u_j}^{LSA} * s * profile_{i_k}^{LSA} \tag{7.7}$$

The output of this phase is the list of items sorted by their corresponding $p_{u_j,i_k,C}$. The top items are selected to compose the recommendation.

7.3 Experimentation and evaluation.

This section describes the experiment performed to evaluate LSAContextCluster in various contexts. First, the experimental procedure is explained. After that, the techniques compared are detailed. Later, the datasets used in the experiment and the methods for processing them are described. Afterwards, the evaluation measures used in the experiment are detailed. Finally, the results are shown and analyzed.

7.3.1 Experimental procedure

The procedure proposed by Sarwar et al. [203] was followed with certain modifications to consider contextual information:

- Split the dataset in training and test.
- Build the model from the training data.
- Build the profile of each user including contextual information if applicable.
- Recommend to each user based on their profile and the model.
- Evaluate recommendations with the test set.

7.3.2 Techniques compared

The baseline method to compare with was the LSA method without contextual information. For the sake of fair comparison, the number of features is fixed to 30 features in all the techniques compared in this experiment.

The proposal is evaluated considering three ways to characterize the context and include it in the QA recommendation:

- No clustering (LSAContext): The topic in the context profile is considered to be unique. Therefore, a single profile of the context is built combining the profiles of all terms:

$$profile_C = \sum_{t_j \in C} * profile_{t_j} \tag{7.8}$$

- Weighted by membership (LSAContextClusterFuzzy): Fuzzy c-means clustering is applied. Each cluster profile is calculated from the terms included in the target cluster. The term profiles are weighted by their membership to the cluster:

$$profile_{c_l} = \sum_{t_j \in c_l} \mu_{t_j,c_l} * profile_{t_j} \tag{7.9}$$

  where μ_{t_j,c_l} is the membership of term t_j to cluster c_l.

- Max membership (LSAContextClusterMax): Fuzzy c-means clustering is applied. A given term is only used in one topic profile, which is the one to whom it has the highest membership:

$$profile_{c_l} = \sum_{t_j \in c_l} \mu^{max}_{t_j,c_l} * profile_{t_j} \tag{7.10}$$

  where $\mu^{max}_{t_j,c_l}$ is one if μ_{t_j,c_l} is the maximum membership across all clusters, and zero otherwise.

The methods compared have the parameter α which is evaluated exploring the performance of all techniques with $\alpha \in [0.90, 1.00]$ with increments of 0.01.

7.3.3 Datasets

Two sources of data are considered in the proposal: The QA domain and the contextual information. Therefore, two datasets are considered in the experiment.

StackExchange dataset[1] is used as input from the question answering domain. It consists of the database dump of each site in the StackExchange ecosystem. In

[1] `http://data.stackexchange.com/`

Table 7.1: Main stats of some StackExchange site datasets.

	3dprinting	academia	android	history
Users	4025	48448	119810	13433
Questions	597	57967	41423	6127
Answers	1135	16737	49985	12212
Comments	2754	40319	123291	53510
Votes	7860	138416	359710	162809
Ratings	2458	119082	109644	38082
Sparsity	0.99898	0.99996	0.99998	0.99954

this experiment, we focus on the StackExchange site devoted to 3D printing[2]. Context influence might vary across sites, hence we selected the 3D printing site given the current interest on such topic. Table 7.1 shows relevant stats of the dataset.

The contextual dataset was defined through a set of interesting keywords. The proposal focuses on selecting currently hot topics, therefore, the terms *news*, *current* and *situation* were selected. A dataset of tweets containing at least one of these terms was extracted from Twitter, whose stats are depicted in Figure 7.3.

7.3.4 Evaluation Measures

The prediction error is the most widely used measure to evaluate the performance of recommender systems. However, the techniques compared in this experiment do not predict ratings. Therefore, the measures available for this evaluation are information retrieval ones, such as prediction or recall. However, previous researchers [95] have remarked that they are not sensible to the sorting of items. Therefore, the NDCG is used:

$$NDCG = \frac{DCG}{DCG_{perfect}} \tag{7.11}$$

[2] `https://3dprinting.stackexchange.com/`

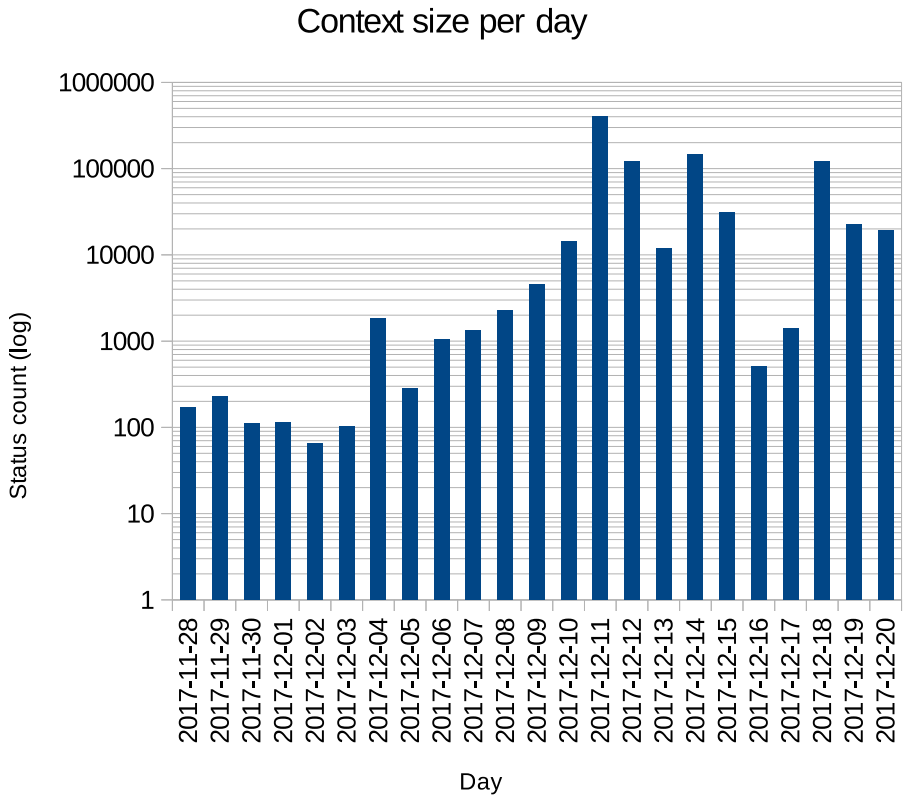


Figure 7.3: Contextual dataset used in the experiment, where each day has a different status update count.

where $DCG_{perfect}$ is the best sorting, this is, the items sorted by their rating in the test set.

$$DCG_u = \sum_{k=1}^{N} \frac{r_{u,recom_{u,k}}}{log_2(k+1)} \tag{7.12}$$

where $recom_{u_j,k} \in I$ is the item recommended to user u in k position.

The dataset is split in training and test sets using the 5 cross-fold validation, which was executed 20 times to generate different partitions and the results on each of them were averaged.

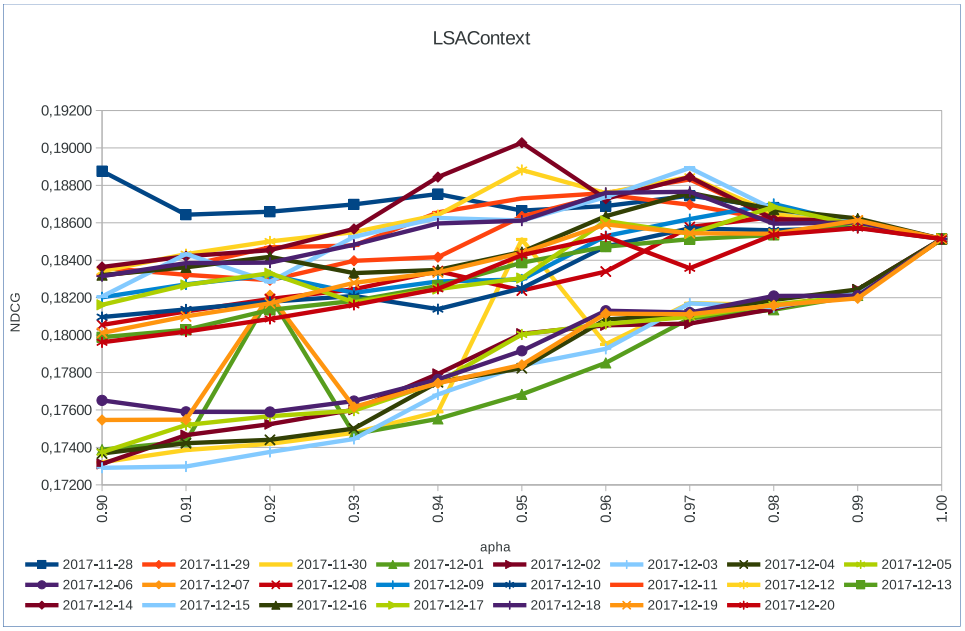


Figure 7.4: Results managing context without clustering

7.3.5 Results

Figures 7.4, 7.5, and 7.6 show the results of the techniques compared. X axis represents the alpha parameter value. Y axis represents the NDCG. It is worth to note the $\alpha = 1.00$ result, which corresponds to the baseline approach because it ignores the context. In these figures, each series denote the context, whose position shows the results of the proposal with the corresponding *alpha* value for the specific day.

Figure 7.4 shows that LSAContext improves the results of LSA ($\alpha = 1.00$) in some days (contexts). However, the improvement on these days does not compensates the loss of performance in others. Specifically, LSAContext improves the results of LSA for all alpha values in 2017-11-28.

Figure 7.5 shows that LSAContextFuzzy improves the results of LSA ($\alpha = 1.00$). As compared to LSAContext, it obtains better results because LSAContextFuzzy results are distributed higher. The results also show that LSAContextFuzzy results improved greatly in 2017-11-28. However, In 2017-11-29, 2017-11-30 and 2017-12-15 there is a greater magnitude of loss in performance, where

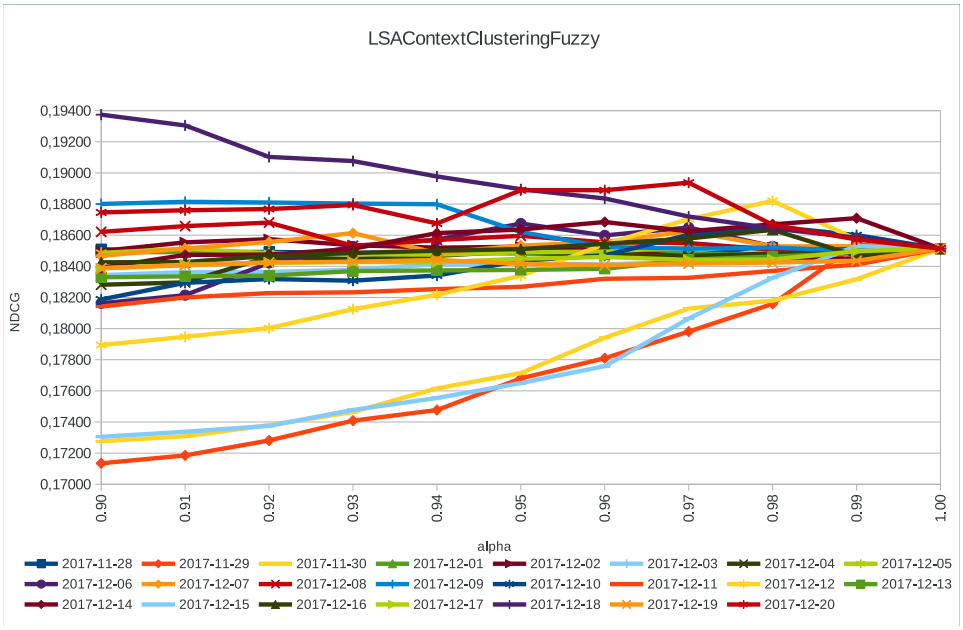


Figure 7.5: Results managing context with fuzzy membership.

LSAContextFuzzy does not overcome the results of the baseline. In 2017-12-12, $\alpha \in [0.90, 0.95]$ yields less performance, but $\alpha \in [0.96, 0.99]$ provides better results as compared to LSA.

Figure 7.6 shows that LSA ($\alpha = 1.00$) results are improved by LSAContextClustering for the majority of the contexts. In some contexts, the good results obtained are cancelled by the worst results in others. However, LSAContextClustering results are consistently better than those of LSA.

The results of each proposal with the best alpha value are shown in Figure 7.7. Notice that LSA does not consider the context, therefore, it has no variability across days. The results of LSAContext across days have a great variability. The better results for LSAContext are from 2017-12-08 onwards, but its performance from 2017-11-30 to 2017-12-07 is much worst. On the other hand, the performance of LSAContextClusteringMax and LSAContextClusteringFuzzy has no drastic drop in any context. Their performance show ups and downs with no clear best approach between them. LSAContextClusteringMax results achieved greater peaks than those achieved by LSAContextClusteringFuzzy in some contexts.

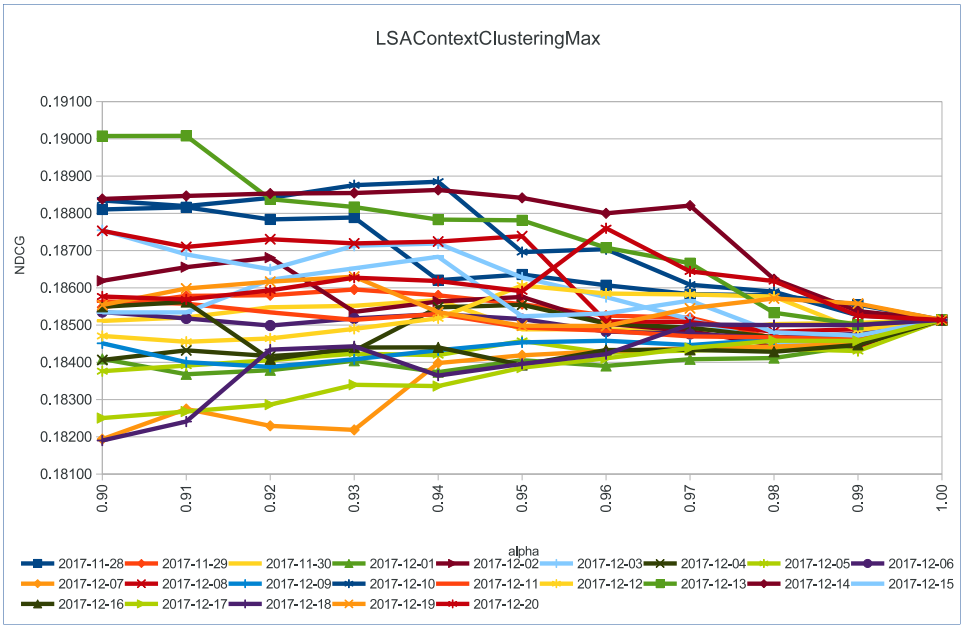


Figure 7.6: Results managing context with max membership.

Figure 7.8 shows the average performance of each approach across contexts. LSAContextFuzzy and LSAContext obtain better results than LSA in certain contexts but in average they do not overcome LSA results. On the other hand, LSAContextClustering results overcome those obtained by the remaining approaches for $\alpha \in [0.90, 0.97]$. Specifically, the best NDCG was obtained bt LSAContextClustering for $\alpha = 0.94$.

The evidence gathered in the experiment shows that the best technique is LSAContextClustering with $\alpha = 0.94$. In other domains, the optimum value of parameter α might vary given that this value has been optimized for the 3dprinting QA dataset.

7.4 Summary

This chapter presents a proposal for the integration of current trend interest as the context of question answering recommendation. Specifically, the model proposed applies first semantic analysis of QA domain, analyses the context applying clustering and contextualizes user's profiles with the analysis of the context. This

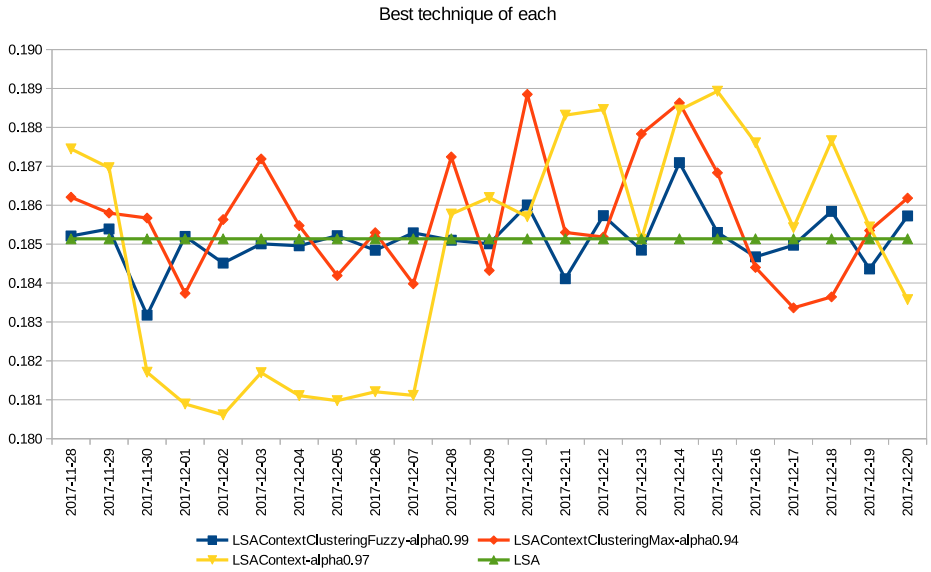


Figure 7.7: Results of the proposal to manage context with fuzzy membership.

process allows to integrate current trend interest in the question answering recommendation. The results of the experiment show that the integration of contextual information extracted from current trend interest improves QA recommendation. Specifically, the best way to generate the context model is to build the context topic profile using only the terms whose membership value to the target cluster is the highest.

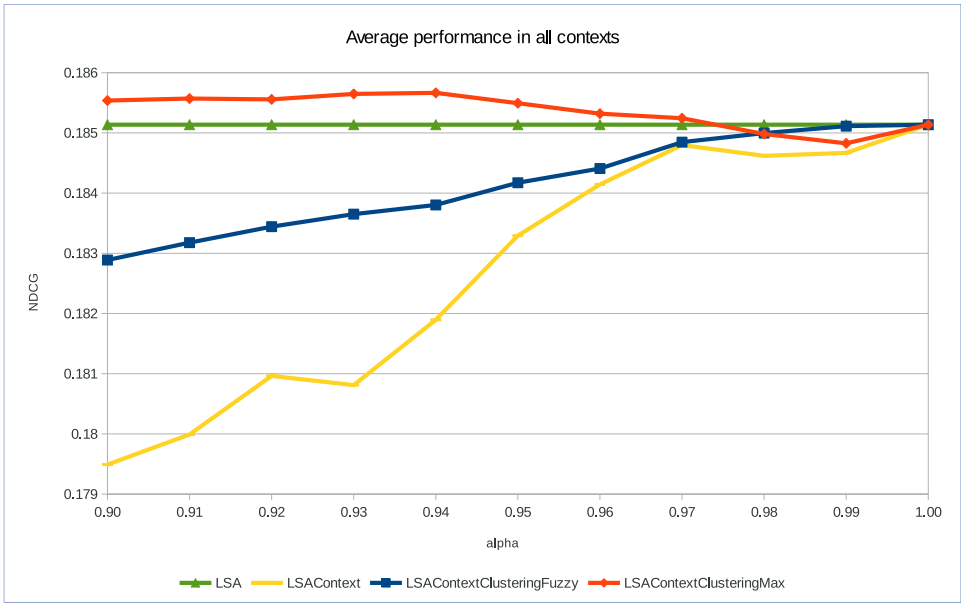


Figure 7.8: Average NDCG of the compared approaches in all contexts.

Chapter 8

REJA: Location-aware Consensus-driven Recommender System for Restaurants

8.1 Introduction

In the previous chapter, a model to deliver question answering context-aware recommendations based on the current trend interest extracted from a microblogging system was proposed. The integration of contextual information enhanced the performance of question answering recommendation.

The high interest of people in traveling for leisure has increased the amount of alternatives for tourists, particularly in areas with relevant natural environment, monuments or tourist attractions in general. The decision of what to do is affected by information overload, specially if we focus on the case of restaurants. In these places, the increasing amount of offers sometimes makes tourists overwhelmed. Often, the selection of a restaurant leads to sub-optimal decisions, given that users have limited time to choose a restaurant that meets their preferences and needs. Therefore, recommender systems can be used in order to support users in such a decision. Moreover, the importance of context-awareness [88, 94, 191] and group recommendation [222] for mobile restaurant recommender systems has been highlighted in previous research. This motivates us to extend REJA [140, 161, 195], a restaurant recommender system for the province of Jaén developed in the Univer-

sity of Jaén (Spain), to integrate context-aware and group recommendations. Consequently, this chapter presents two prototypes that implement (i) a location and trajectory-aware recommender system, and (ii) a location-aware consensus-driven group recommender system.

The first prototype, LT-REJA, integrates contextual features in restaurant recommendation. Context-aware recommender systems (CARS) [6] consider that contextual features influence recommendation utility for users. Hence, CARS focus on the integration of contextual features in the personalized recommendation. In the prototype developed, the *location* is considered to be a relevant contextual feature, given that it poses an important constraint to the utility of restaurants recommendation.

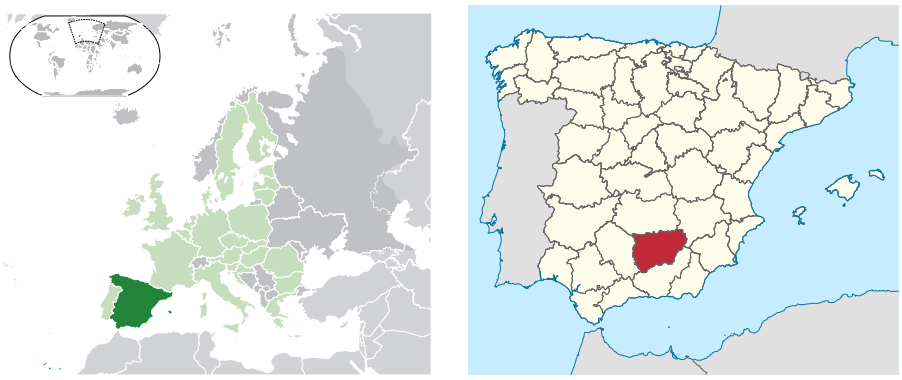
In addition to the integration of contextual features, it is also interesting to consider group recommendation and integrate group agreement to improve the group recommendations. With such an aim, CLG-REJA extends REJA to provide context-aware consensus-driven group recommendations.

The remainder of this chapter is structured as follows. Section 8.2 describes REJA. Section 8.3 introduces the prototype to provide location and trajectory-aware recommendations LT-REJA. Section 8.4 presents the prototype to provide location-aware consensus-driven group recommendations. A summary is provided in Section 8.5.

8.2 Restaurants of Jaén Recommender System: REJA

There are various widespread systems to search and check restaurants, such as Yelp or TripAdvisor. However, location specific applications are able to provide an added value that generalistic applications are not able to deliver. REJA[1] (REstaurants of JAén) [140, 161, 195] is a recommender system developed by Sinbad[2] Research Group at the University of Jaén (Spain). REJA is a location specific application that focuses on the recommendation of restaurants located in the province

[1] `http://sinbad2.ujaen.es/reja`



(a) Spain (b) Province of Jaén

Figure 8.1: Province of Jaén, location of interest of REJA

of Jaén.

Some relevant features of the environment of REJA are provided. The population in the province of Jaén is 654,170 (2015), which is distributed in 14,496km^2. The production of olive oil is the most important economic activity, which occupies around 80% of the cultivable land. Regarding its tourism features, it is worth to remark that there are four nature parks. There are more than eighty castles and fortresses in various towns which are in a good preservation state and many of them were declared *Bien de interés cultural* (good of cultural interest). Several renaissance monuments are also well preserved, such as churches and palaces. These features made the province of Jaén a location of interest for tourists, and there are a number of restaurants distributed along the province.

REJA was developed to support users that want to find restaurants that satisfy their preferences and needs in the province of Jaén through the delivery of personalized recommendations. User feedback is provided as explicit ratings over the restaurants. The database is composed of 516 restaurants, and the information that the system holds about a restaurant is composed of location, phone number, and type of cuisine, among other relevant features regarding restaurant facilities.

REJA is also able to provide recommendations for anonymous users through some non-personalized recommendation techniques, such as the most-liked and

the most-popular ones. It is also able to provide a list of restaurants that are similar to a given one.

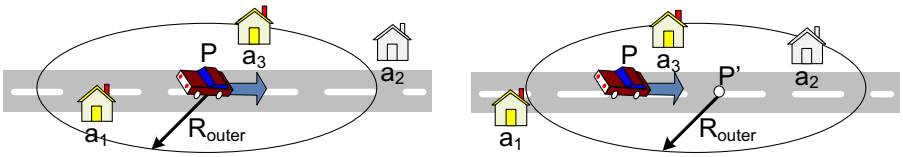
Regarding the personalized recommendations, REJA applies the user-based collaborative filtering. A registered user that wants to receive recommendations must provide a minimal set of ratings in order to state their preferences. With this information, REJA builds a user's profile that is later used to compute personalized recommendations.

A common problem of collaborative filtering recommender systems is the availability of information regarding users. The recommendation for users with a small amount of information cannot be computed or it is low quality. Such a problem is named user cold-start. REJA is able to provide recommendations for such users with the implementation of a commuted hybrid recommender system [195], which uses a knowledge-based recommender system in cold-start cases.

8.3 Location and trajectory-awareness for recommendations on the move.

The aforementioned functionality of REJA was developed to be used from home through a web interface. However, most of interaction with such systems has recently changed to mobile devices. Interaction from mobile devices has several limitations, such as screen size, non-physic keyboard or battery duration. On the other hand, mobile devices are equipped with several sensors, such as barometer, accelerometers, wireless communication interfaces, compass and global positioning system (GPS), which are a source of information about the context in which the interaction with them is done. For this reason, mobile devices interaction is a good candidate to integrate valuable information of users context, which make the integration of context-aware recommender systems easier. This later feature of mobile devices allows to extend REJA recommendations to enable location-aware requirements.

An interesting case of use of REJA is the following: a user is traveling along the



(a) AOI centered on the target user's location.

(b) AOI displaced to consider speed and trajectory.

Figure 8.2: Displacing the AOI according to the speed and the user-defined parameter R_{outer}.

province and driving to the next destination, but he wants to stop in a restaurant in the way. For that, the user has already stated how far is willing to travel through the R_{outer} parameter. If the recommender system considers an area of interest (AOI) centered in the user's location (see Figure 8.2a), then the system would recommend restaurants that are already behind user's path to the destination, which is not useful. Instead, recommendation utility would increase if the system considers contextual features to avoid recommending items that are already left behind. The mobile device is capable of providing location, but also speed and trajectory. Therefore, the proposed model adjusts the target user's area of interest displacing it to better reflect user's context (see Figure 8.2b). This adjustment and its application to filter relevant recommendations results in a contextual post-filtering that adjusts the recommendations to the target user context.

Therefore, a location and trajectory-aware recommendation model is proposed. The architecture of the model applies contextual post-filtering [6] in two phases (i) context-aware filter; and (ii) traditional recommender system. This model uses location, trajectory and speed of the target user in order to adjust the recommendations to increase utility.

In the context-aware filter phase, the target user's area of interest is calculated to be later applied to the recommendations. With such an aim, the model holds the user's location P. In order to adjust the AOI, the P' point is calculated, which is a translation of the users' location regarding the speed and trajectory. With such endeavor, the system holds the historic of positions of the target user. This

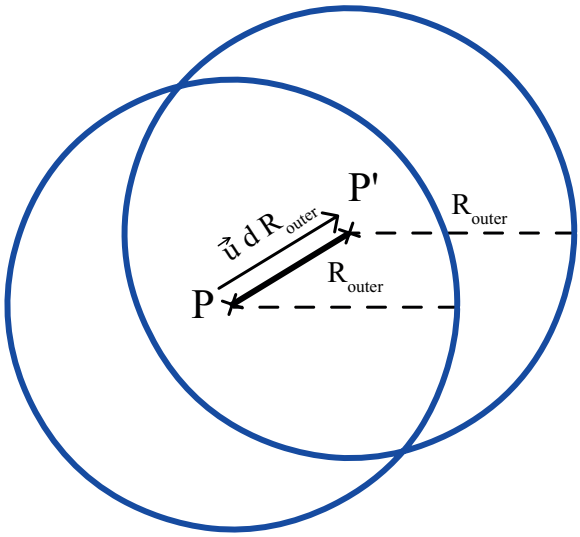


Figure 8.3: The AOI is translated from P to P', according to the user's speed and trajectory.

information is user together with the values given by the GPS to determine user's speed and trajectory, which determine the transformation T applied to P to obtain P' (see Figure 8.3):

$$P' = P + T \tag{8.1}$$

$$T = \overrightarrow{u_j}\ d\ R_{outer} \tag{8.2}$$

where P is the user location, T is the displacement, $\overrightarrow{u_j}$ is a normalized vector that indicates user u trajectory, R_{outer} is a user defined parameter and $d \in [0,1]$ is a fuzzy parameter calculated regarding user's speed:

$$d = \begin{cases} 0 & if \quad speed < a \\ \frac{x-a}{b-a} & if \quad speed \geq a \ and \ speed < b \\ 1 & if \quad speed \geq b \end{cases} \tag{8.3}$$

where *speed* is the average speed of the user in km/h. Considering that speed limit is 120km/h in most countries, we propose to use the following parameter

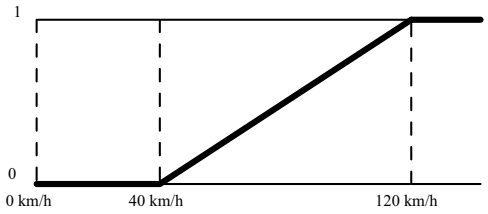


Figure 8.4: d depends linearly on the user's speed

values: a=40km/h, b=120km/h (see Figure 8.4). These values assume that if the user is moving at less than 40km/h, the user is in an urban environment and no displacement of the area of interest is done. In the case of a speed greater than this value, the area of interest moves forward to avoid recommending items left behind. The maximum displacement of the area of interest is reached when moving at 120km/h.

In the recommendation phase, only the items that lie within the area of interest are candidates of the recommendation. Therefore, the recommendation function is modified to consider contextual features:

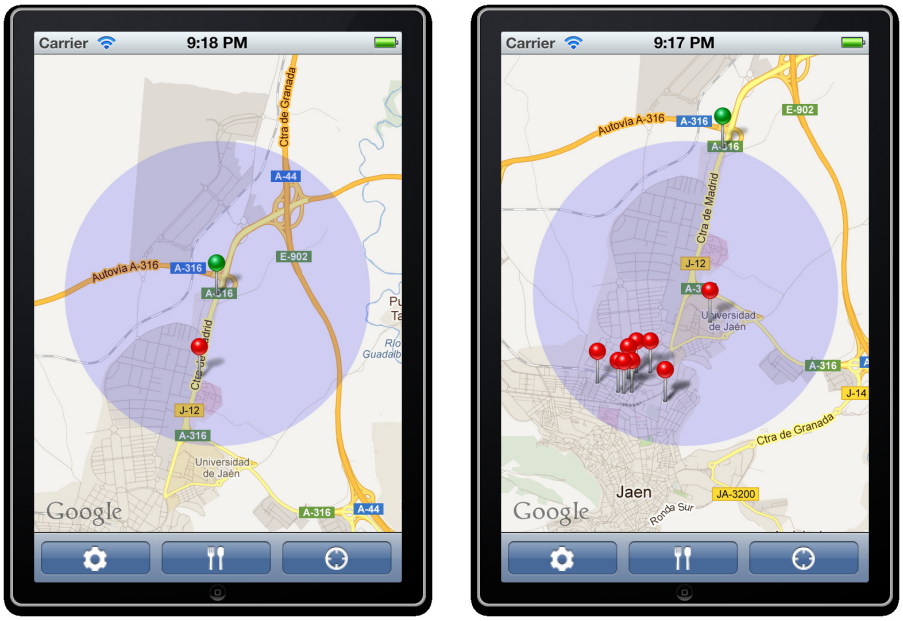
$$Prediction(u_j, i_k, P, d, \overrightarrow{u_j}) = \tilde{r}_{u_j,i_k} * min(R_{outer} - distance(i_k, P'), 0) \qquad (8.4)$$

where $distance(i_k, P')$ is the euclidean distance from the center of the AOI to item i_k location. It is worth to mention that the prediction function is zero for items out of the AOI, therefore the computations needed to recommend are only those of items within the AOI.

Finally, the items with the highest prediction value are recommended to the user.

8.3.1 Prototype: LT-REJA

LT-REJA implements the aforementioned model. Figure 8.5 depicts two examples of use. Figure 8.5a shows the location of the user with a green pin. Given that the user location is fixed, the area of interest is centered in this location. Figure 8.5b



(a) On site request (b) On-the-move request

Figure 8.5: Area of interest for two user's contexts.

shows the area of interest of a moving user, which appears ahead of the path in order to consider only restaurants that are in the way of his movement and overlooks those already left behind.

The prototype has a client-server architecture depicted in Figure 8.6. The client part is composed of the mobile device and the application, which is in charge of the user interaction, gathering the context and requesting recommendations. The server part is composed of the services available to respond to the client queries, which also runs the recommender system.

The prototype requests user log-in data and prompts the user with a slider to provide the desired R_{outer} parameter value. Moreover, the prototype starts gathering the location from mobile device sensors to determine the user's location, trajectory and speed. With this information, the client can compute the adjusted AOI center P' and use it later to request recommendations.

When the server receives a recommendation request, it selects the set of candidate items comparing all items location to the center of the adjusted AOI sent by

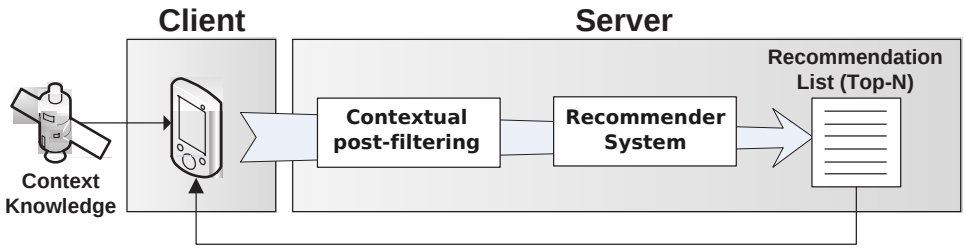
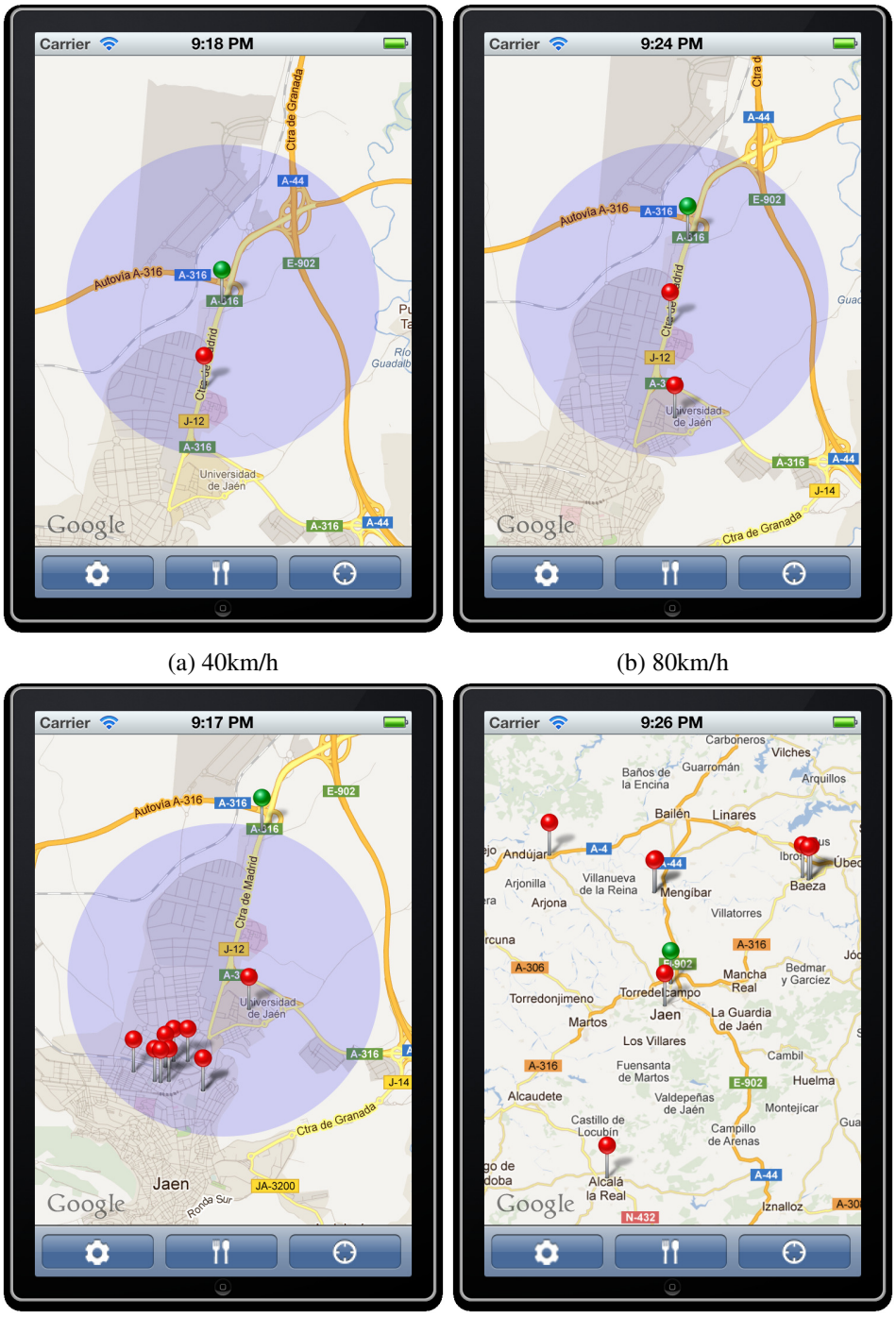


Figure 8.6: LT-REJA prototype architecture. Contextual information is incorporated with the contextual post-filtering approach.

the mobile device. This comparison results in a reduced list of items for which the server needs to predict a rating value, which is the computational expensive part. When this process finishes, the server side responds with a XML that contains the recommended items with their descriptive information, such as restaurant location, description, phone, among other details.

The recommendations are depicted on a map-based interface. The client part is in charge of requesting new recommendations to the server whenever the user's context changes and the adjusted area of interest changes. This way, the recommendation list is updated as the user travels.



(a) 40km/h (b) 80km/h

(c) 120km/h (d) No context.

Figure 8.7: LT-REJA prototype screenshots for various contextual situations. User's location is depicted with the green pin, and recommended restaurants with red pins. The adjusted AOI has been depicted in these figures with a blue circle (R_{outer}=2km).

Figure 8.7 illustrates the behavior of the prototype for several contextual situations. Figures 8.7a, 8.7b, and 8.7c depict the adjusted area of interest P' calculated by the mobile device with a green pin and the red pins are the recommended items. Notice that for all these cases, the location and trajectory of the users is constant, and the speed is 40km/h, 80km/h and 120km/h, respectively. Notice that the blue circle is not shown in the prototype, it has been added here for the sake of clearness. Figure 8.7d shows the recommendations computed without considering the context, which shows recommended items spread all over the province of Jaén.

8.4 Location-aware consensus-driven group recommendation

Previous sections describe the functionality of REJA regarding recommendation, such as non-personalized recommendation, knowledge-based recommendation, or the integration of contextual features. Restaurants are social items in the way that they are generally used in groups, therefore, a restaurant recommender system should provide group-personalized recommendations. This section describes the integration of context-aware features in the consensus-driven group recommender system. This extension aims to provide group-personalized recommendations that satisfy group's members and are also suitable regarding the context in which they are.

In REJA, the feature of the group context considered relevant is the location of the group. Regarding the contextual information available in the database over the items, the location of the candidate restaurants is stored. Given the limited amount of feedback available in REJA database, pre-filtering approaches are not used because they suffer from data sparsity and large amount of information is needed in order to overcome the results of post-filtering contextual integration approaches [7]. Contextual post-filtering allows to apply a re-rank or filter function to recommended items in order to discard those recommendations that are not relevant regarding the context in which the recommendation is requested. The context

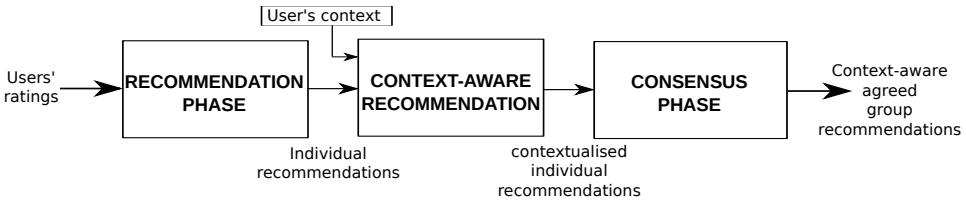


Figure 8.8: General scheme of the context-aware group recommender of REJA

considered in REJA is composed of the location of the target group, therefore, a function that penalizes items far from the group is applied in the post-filtering.

ConsensusGRS does not consider contextual information as is defined in Chapter 4, therefore, a step to contextualize the recommendation is included in order to provide the model with location-awareness. Figure 8.8 depicts the general scheme for the consensus-driven location-aware group recommender system, which is composed of the following phases:

1. Individual recommendation: A single user RS is used to generate members' individual recommendations.

2. Recommendation contextualization: A post-filtering function is applied to consider group context in the recommendation and produce contextualized individual recommendations.

3. Consensus phase: The automatic consensus module processes the contextualized individual recommendations to compute the contextualized agreed group recommendations.

These general phases are detailed in the remaining of this section.

Individual recommendation

In the individual recommendation phase, the individual recommendations are generated. The single user recommender system computes recommendations for each member individually. A list of items with their rating prediction is generated for

each member. These recommendations lists are filtered to consider only items that are commonly recommended to all group's members.

Recommendation contextualization

Once the individual recommendations are computed, a contextual post-filtering process is applied in the recommendation contextualization phase in order to consider group's location. Therefore, a distance to the group location is computed in order to re-rank items.

A fuzzy method is applied in this re-ranking in order to provide certain flexibility in the process. This flexibility support the election of parameter δ by the group manager, which reflects the distance that the group considers suitable to move in order to reach a restaurant that suits their preferences. The parameter δ is used by the system to discard those items that are too far to reach, and the prediction of items that lie near the boundaries of the area of interest is modified in a soft way according to Eq. 8.5 and 8.6.

$$\tilde{r}'_{m_j i_k} = r_{m_j i_k} * w_{G,i_k} \qquad , w_{G,i_k} \in [0,1], m_j \in G \tag{8.5}$$

$$w_{G,i_k} = \begin{cases} 1 & if \quad d(G,i_k) \leq \delta \\ 1 - \frac{d(G,i_k)-\delta}{\delta'-\delta} & if \quad \delta \leq d(G,i_k) \leq \delta' \\ 0 & if \quad d(G,i_k) \geq \delta' \end{cases} \tag{8.6}$$

where $d(G,i_k)$ is the distance between group and item location, δ is defined by the group manager, and δ' value is defined from δ:

$$\delta' = \delta * (1+\alpha), \qquad \alpha \in [0,1] \tag{8.7}$$

where α is a parameter that defines how flexible is the contextual filtering. In REJA, α is set to 0.2, but it might be different in other recommendation domains.

After this, the contextualized predictions are first converted to order relations $O_{m_j i_k}$ ranking items by rating prediction in descending order. Similarly to the model described in Chapter 4, the items are pre-filtered in order reduce the set of alternatives for the later consensus reaching process. This is done with the single transferrable voting [74], which outputs the set of items considered in the recommendation $\tilde{I}_{\tilde{R}_{G\bullet}}$.

Order relations $O_{m_j i_k}$ are first filtered to keep only items in $\tilde{I}_{\tilde{R}_{G\bullet}}$ and generate $\tilde{O}_{m_j i_k}$. After that, filtered order relations $\tilde{O}_{m_j i_k}$ are transformed to preference relations using Eq. 4.5. The results of this phase are the fuzzy preference relations P_{m_j} of all member $m_j \in G$, which are passed to the consensus phase.

Consensus phase

The fuzzy preference relations of all members are used in the consensus phase as the initial preference of members regarding the items. The automatic consensus support system model based on feedback [170] is applied over them, as explained in Section 4.2.2. The items with the highest preference value for the group are recommended.

8.4.1 Prototype: CLG-REJA

LT-REJA implements the aforementioned model. The prototype aims at providing restaurant recommendations in the province of Jaén. Figure 8.9 depicts the architecture of the prototype, which follows the client-server architecture paradigm.

The mobile devices of group's members compose the client side, which have the prototype installed. The prototype allows group's members to create the group in the system and join. The client side gathers contextual information of the users when they request recommendations. The server side provides a web service that allows to gather users' preferences, holds information about the group composition and processes the contextual information sent by the client side. With this information, it computes the recommendation and sends it back to the clients, who are in charge of displaying it.

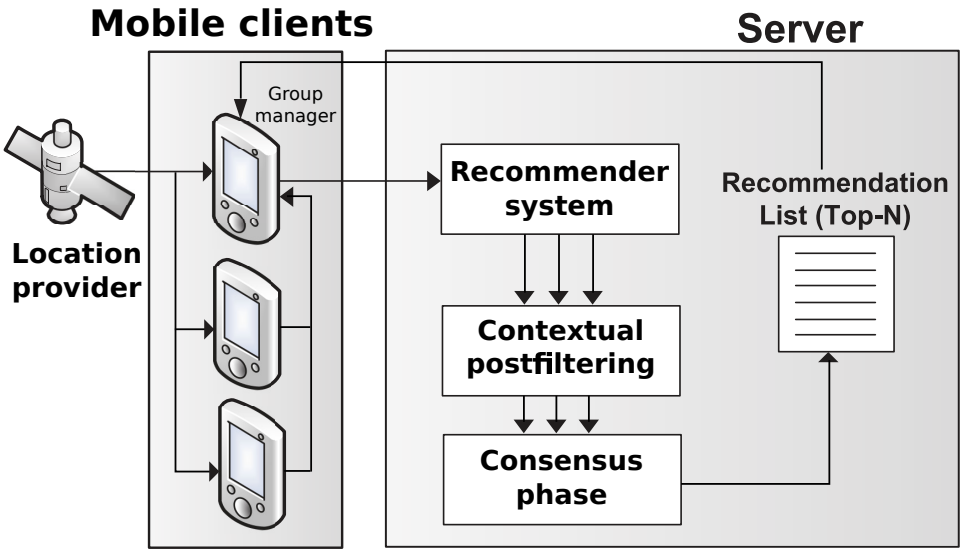
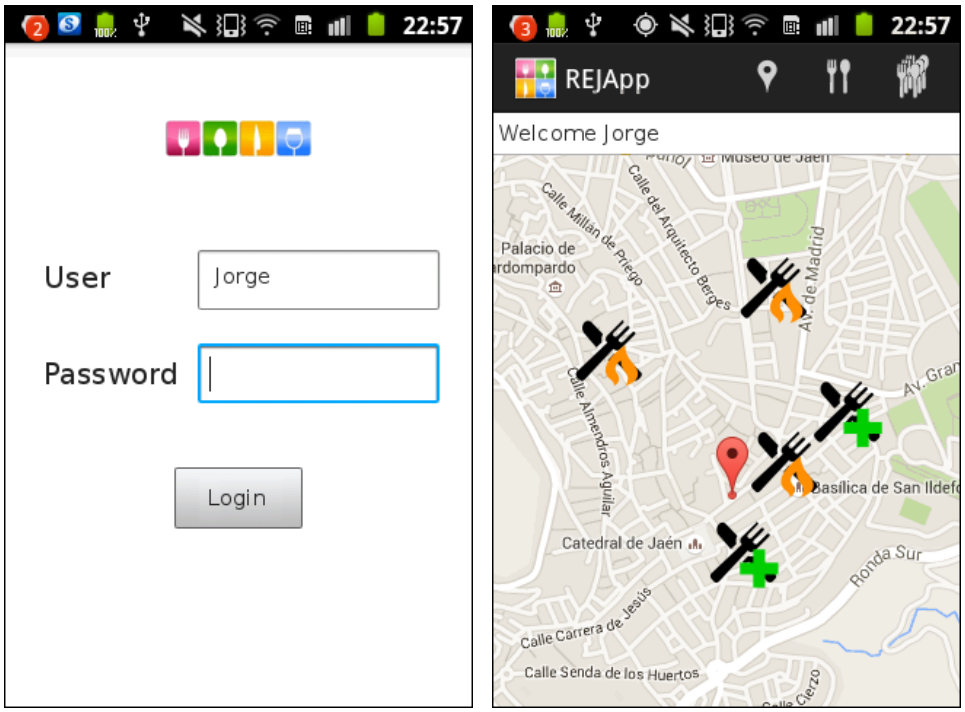


Figure 8.9: Architecture of the prototype for location-aware consensus-driven group recommendations.

Therefore, the client-side prototype is installed on users' devices. Once the application is launched, the users provide their log-in data. Figure 8.10 depicts the screens used for the log-in task. Specifically, the login screen is depicted in Figure 8.10a, and the initial screen after the login is shown in Figure 8.10b

After all group's members have completed the login task, a user assumes the role of group manager creating the group and sending join invitations to the remaining members. The aim with this task is to build the group and send this information to the server side. Figure 8.11 shows the create group task. Specifically, a screen with the groups already created is depicted in Figure 8.11a, which also allows to create a new group. The screen to add group's members to the new group is shown in Figure 8.11b.

When the group is created, the group creator is able to request recommendations for the groups as shown in Figure 8.12a. This task is composed of two screens. Figure 8.12a shows the screen provided to gather groups preferences regarding the recommendation. This screen provides a slider to allow the group manager to pick the desired value for δ. The group recommendations are displayed in a



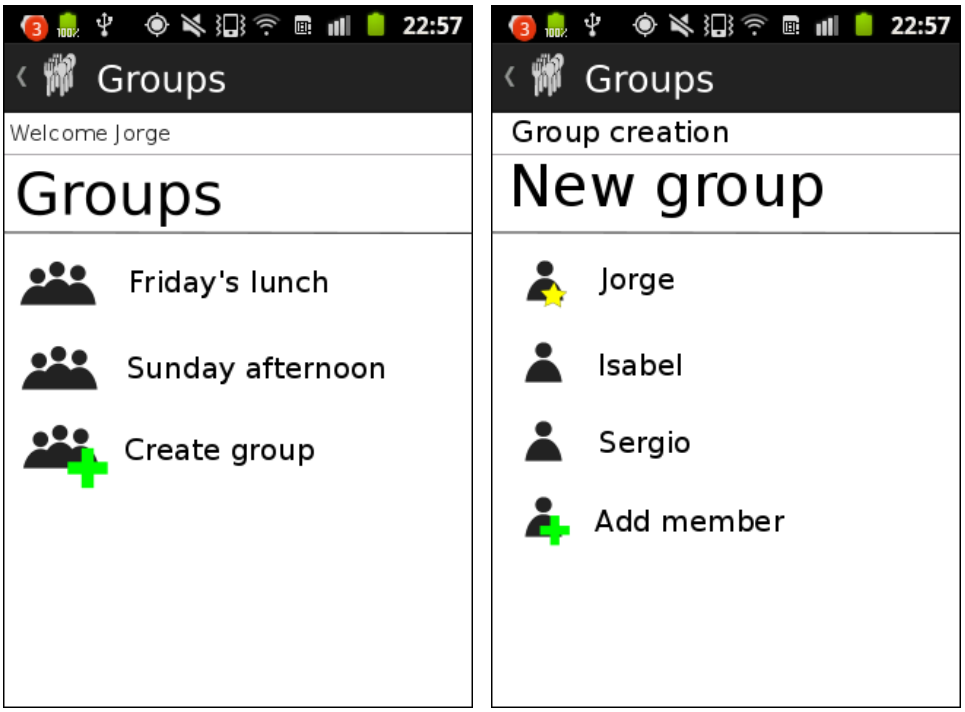
(a) Login page. (b) Logged user screen.

Figure 8.10: Screens of the prototype for the login task.

map centered at the group location as shown in Figure 8.12b, which allows group's members to explore the recommended restaurant in details. This map visualization supports the group at taking the final decision showing them relevant information of the restaurants recommended.

8.5 Summary

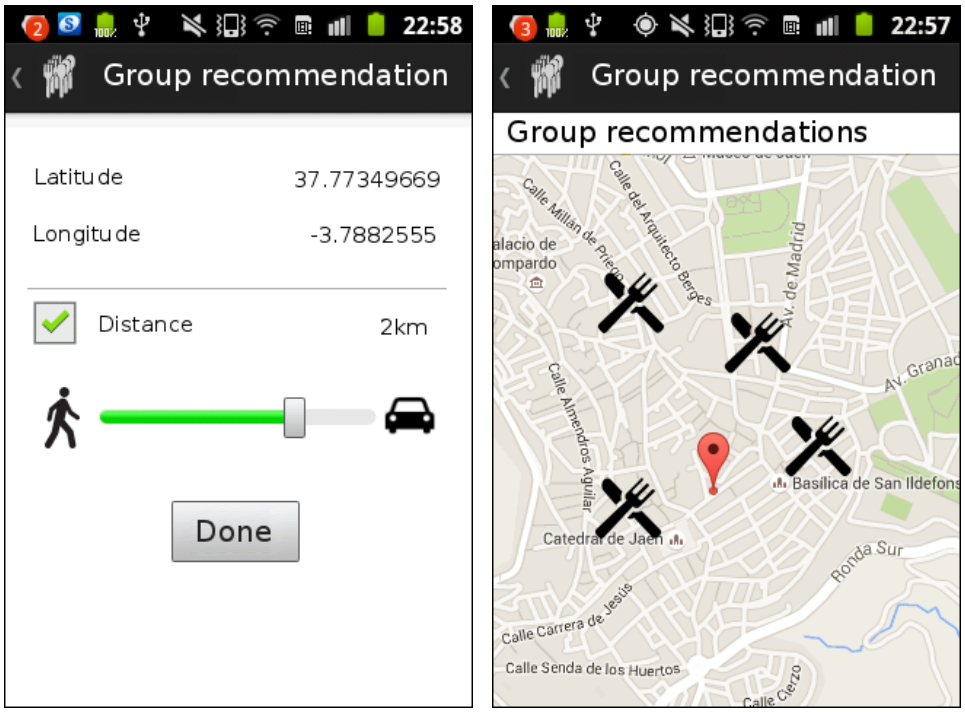
This chapter has shown two models developed to support the recommendation of restaurants. The first model integrates contextual information to the restaurant recommendation for the case of use in which a user request recommendations when traveling. Therefore, the recommendation model considers the location and trajectory to recommend items ahead of the user's position. This model is implemented in a working prototype named LT-REJA. The second prototype considers that restaurants are social items, therefore, they are recommended to groups.



(a) Group creation. (b) Members specification.

Figure 8.11: Screens of the prototype for the group formation task.

Given that the satisfaction of users needs is a key aspect in the recommendation of restaurants, the consensus-driven model proposed in Chapter 4 is extended to provide context-aware recommendations through the consideration of group location. Therefore, the general recommendation scheme of the consensus-driven GRS is modified applying contextual post-filtering to re-rank alternatives regarding their suitability to the recommendation context. This modified model is implemented in a working prototype named CLG-REJA. Both prototypes extend REJA capabilities to be used in mobile devices.



(a) Recommendation request. (b) Map visualization.

Figure 8.12: Screenshots of prototype for the recommendation task.

Chapter 9

Conclusions and further study

This chapter concludes the whole thesis and provides some further research directions of the topic.

9.1 Conclusions

This research is motivated by the challenges and practical new trends in recommender systems such as group recommender systems and context-aware recommender systems. The main contributions on this thesis are the following ones:
1) It develops a hesitant fuzzy sets-based representation of group's preferences (Research Objective 1) and a hesitant fuzzy sets-based group recommendation approach (Research Objective 2) in Chapter 3.

A method to represent group's preferences using hesitant fuzzy sets is proposed, where the multiple ratings stated over one item by all members are modeled as the group hesitation regarding the rating of such an item. This way, the aggregation process is avoided and also the associated information loss. A novel recommendation method called HGRM that builds upon the aforementioned group modeling is developed. The group's preference modeling with hesitant fuzzy sets is used to find suitable neighbors to the target group. The hesitant Pearson's correlation coefficient allows to compute similarities between groups and users to build the group neighborhood. This way of computing the similarity to the group avoids to aggregate the group ratings in a pseudo user, which overlook specific

features of the group ratings and diminishes the diversity of the ratings. The experiment performed shows that HGRM enhances recommendation diversity as long as group size increases. A mixed metric of *accuracy* and *diversity* is used to combine both, for which HGRM shows remarkable improvements for different relative importance of accuracy over diversity. This evidence suggests that the new proposal delivers more diverse and accurate recommendations compared with other approaches.

2) It develops a framework for group recommendation based on consensus reaching processes and two models that integrate the minimum cost consensus model (Research Objective 3) and the automatic consensus support system model based on feedback (Research Objective 4) in Chapter 4.

A group recommendation framework for consensus reaching processes is developed. Two models within this framework have been developed. The first model applies the minimum cost consensus model to find the minimum changes that increase the consensus within the group. The second model applies the automatic consensus support system model based on feedback to improve members agreement before the recommendation. In both models, the group recommendation is calculated after a high agreement is reached. This framework provides existing group recommender systems with a way of improving members' satisfaction regarding the group recommendation. The experiments performed suggest that the proposed framework for consensus in group recommendation improves the results compared with other approaches.

3) It develops an extension of opinion dynamics model dealing with relationships among members for group recommendation (Research Objective 5) and a group recommendation approach that ensures consensus on members opinions (Research Objective 6) in Chapter 5.

The Pre-GROD model is developed, which is an extension of the DeGroot's opinion dynamics model for group recommendation. The framework considers the relationships between members' preferences in the group recommendation with

DeGroot's model, which is used to reproduce the opinion change within the group regarding the relationships among members' preferences. Moreover, an extension of the DeGroot's model for group recommendation named GROD is developed to ensure that the group always reaches consensus. In the cases where the opinion evolution in the DeGroot's model does not lead to consensus opinions, a modification of the members' relationships is done to ensure that opinions always reach a consensus value. The experiments performed show that both Pre-GROD and GROD overcome the baseline. In one experiment, Pre-GROD is evaluated to determine the best similarity measure, which shows that asymmetric similarities play an important role in the analysis of members' preferences. This fact suggests that asymmetry reflects better how the group makes decisions. In a second experiment, GROD model is evaluated in groups without consensus, which shows that ensuring consensus improves recommendation compared to the baseline and Pre-GROD. In conclusion, opinion dynamics models and the consideration of consensus within them provides better recommendations as compared with other approaches.

4) It develops extensions of natural noise management focused on group recommender systems databases (Research Objective 7) and a pre-processing approach for natural noise management in group recommendation (Research Objective 8) in Chapter 6.

A natural noise management approach for group recommender systems is developed. The specific features of the group recommendation show two levels in the ratings: the local level regarding the group and the global level. Three approaches are developed considering these levels: NNM-LL, NNM-LG, NNM-GG. A natural noise management approach for group recommender systems databases named NNM-H is developed building upon the aforementioned approaches through a cascade hybridization. First, a global natural noise management process is performed to reduces the noise in the entire dataset. After that, a refinement step focuses on target group ratings. The experimentation shows that the application of NNM in group recommendation provides improvement to a group recommender system

that uses the de-noised ratings dataset.

5) It develops a method for topic identification within collaborative trend interest (Research Objective 9) and a context-aware recommendation approach considering collaborative trend interest (Research Objective 10) in Chapter 7.

A method for detecting the multiple topics in context extracted from microblogging systems is developed to separate the several topics that are present in users status updates. This is done applying the fuzzy c-means clustering algorithm, which improves context characterization. A method for semantic context-aware question answering recommendation called LSAContextCluster is developed. This method uses the aforementioned context characterization method to contextualize target user preference profile, which provides both context-aware and personalized recommendations. The combined profile allows the system to consider user's preferences and contextual information in the recommendation. The experiment results show that LSAContextCluster provides better results as compared to the baseline method. The results suggest that the consideration of contextual features benefits the recommendation in community-based question answering.

6) It develops two prototypes for the extension of a tourism recommender system with context-awareness in Chapter 8. One prototype implements a model that integrates a location and trajectory-aware recommender system. Another prototype implements a location-aware consensus-driven group recommender system (Research Objective 12).

Two models are extended to provide context-aware recommendations. The first model integrates location and trajectory-awareness in recommendation. The second model integrates location-aware consensus-driven recommendations for groups. Both models are developed in two working prototypes that build upon the REJA recommender system. Two important issues within tourism are considered, which are the ubiquity and social feature that involves tourism activities. The prototypes developed extend the functionality of REJA to integrate both context-aware and group features in the recommendation.

9.2 Further study

There are still some limitations of the current study:

- There are several models proposed for group recommender systems that address limitations of the previous models, such as the modeling of group's preferences, the consideration of consensus in group recommendation, the integration of group dynamics in the recommendation model and the management of natural noise. These models address the aforementioned limitations separately, and a comprehensive model that combines all these improvements should be provided.

- The proposed model for context-aware question answering recommendation is targeted to individual users. However, there are domains in which group recommendation would be necessary, such as e-learning. Moreover, the characterization of the trend interest is limited to content analysis, which could be improved.

The aforementioned limitations might be overcome with studies in the following directions:

- The combination of the models proposed in this research will be investigated. With such an aim, the hybridization of the approaches is a promising research direction to provide a comprehensive model that delivers diverse recommendations, provides agreed solutions, considers group dynamics and applies noise reduction techniques.

- The extension of the context-aware question answering recommendation model to provide group recommendations will be investigated. Moreover, the characterization of trend interest could be improved considering techniques from social network analysis.

Bibliography

[1] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.

[2] Gregory D. Abowd, Christopher G. Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. Cyberguide: A mobile context-aware tour guide. *Wireless Networks*, 3(5):421–433, 1997.

[3] Pedro Henriques Abreu, Daniel Castro Silva, Fernando Almeida, and João Mendes-Moreira. Improving a simulated soccer team's performance through a memory-based collaborative filtering approach. *Applied Soft Computing*, 23:180–193, 2014.

[4] Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen, and Alexander Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems*, 23(1):103–145, 2005.

[5] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, (6):734–749, 2005.

[6] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender Systems Handbook*, chapter 3, pages 217–253. Springer US, 2011.

[7] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender Systems Handbook*, chapter 3, pages 217–253. Springer US, 2011.

[8] Gediminas Adomavicius and Alexander Tuzhilin. *Context-Aware Recommender Systems*, pages 191–226. Springer US, 2015.

[9] Gediminas Adomavicius and Alexander Tuzhilin Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. Data Eng.*, 17(6):734–749, 2005.

[10] Charu C. Aggarwal. *Content-Based Recommender Systems*, pages 139–166. Springer International Publishing, 2016.

[11] Hyung Jun Ahn. A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, 178(1):37–51, 2008.

[12] Malak Al-Hassan, Haiyan Lu, and Jie Lu. A semantic enhanced hybrid recommendation approach: A case study of e-government tourism service recommendation system. *Decision Support Systems*, 72:97 – 109, 2015.

[13] Xavier Amatriain, Alejandro Jaimes, Nuria Oliver, and Josep M. Pujol. Data mining methods for recommender systems. In *Recommender Systems Handbook*, chapter 2, pages 39–71. Springer US, 2011.

[14] Xavier Amatriain, Neal Lathia, Josep M. Pujol, Haewoon Kwak, and Nuria Oliver. The wisdom of the few: a collaborative filtering approach based on expert opinions from the web. In *Proceedings of the 32nd International ACM SIGIR Conference*, pages 532–539, New York, NY, USA, 2009. ACM.

[15] Xavier Amatriain, Josep M Pujol, and Nuria Oliver. I like it... i like it not: Evaluating user ratings noise in recommender systems. In *User modeling, adaptation, and personalization*, pages 247–258. Springer, 2009.

[16] Xavier Amatriain, Josep M Pujol, Nava Tintarev, and Nuria Oliver. Rate it again: increasing recommendation accuracy by user re-rating. In *Proceedings of the third ACM conference on Recommender systems*, pages 173–180. ACM, 2009.

[17] Sihem Amer-Yahia, Senjuti Basu Roy, Ashish Chawlat, Gautam Das, and Cong Yu. Group recommendation: Semantics and efficiency. *Proceedings of the VLDB Endowment*, 2(1):754–765, 2009.

[18] Liliana Ardissono, Anna Goy, Giovanna Petrone, Marino Segnan, and Pietro Torasso. Intrigue: personalized recommendation of tourist attractions for desktop and hand held devices. In *Applied Artificial Intelligence, Vol 17 (8-9)*, pages 687–714. Taylor & Francis, 2003.

[19] Kenneth J Arrow, Amartya Sen, and Kotaro Suzumura. *Handbook of Social Choice & Welfare*, volume 2. Elsevier, 2010.

[20] Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 03 1997.

[21] Linas Baltrunas, Bernd Ludwig, Stefan Peer, and Francesco Ricci. Context relevance assessment and exploitation in mobile recommender systems. *Personal and Ubiquitous Computing*, pages 1–20, 06 2011.

[22] Linas Baltrunas, Bernd Ludwig, and Francesco Ricci. Matrix factorization techniques for context aware recommendation. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 301–304, New York, NY, USA, 2011. ACM.

[23] Linas Baltrunas, Tadas Makcinskas, and Francesco Ricci. Group recommendations with rank aggregation and collaborative filtering. In *Proceedings of the 4th ACM Conference on Recommender Systems*, RecSys '10, pages 119–126, New York, NY, USA, 2010. ACM.

[24] Riccardo Bambini, Paolo Cremonesi, and Roberto Turrin. *A Recommender System for an IPTV Service Provider: a Real Large-Scale Production Environment*, pages 299–331. Springer US, 2011.

[25] Xinlong Bao, Lawrence Bergman, and Rich Thompson. Stacking recommendation engines with additional meta-features. In *Proceedings of the 3rd ACM Conference on Recommender Systems*, pages 109–116, New York, NY, USA, 2009. ACM.

[26] Josef Bauer and Alexandros Nanopoulos. Recommender systems based on quantitative implicit customer feedback. *Decision Support Systems*, 68:77 – 88, 2014.

[27] Ana Belen Barragans-Martínez, Marta Rey-Lopez, Enrique Costa-Montenegro, Fernando A. Mikic-Fonte, Juan C. Burguillo, and Ana Peleteiro. Exploiting Social Tagging in a Web 2.0 Recommender System. *IEEE INTERNET COMPUTING*, 14(6):23–30, NOV-DEC 2010.

[28] Nicholas J Belkin and W Bruce Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–38, 1992.

[29] Alejandro Bellogín, Iván Cantador, Fernando Díez, Pablo Castells, and Enrique Chavarriaga. An empirical comparison of social, collaborative filtering, and hybrid recommenders. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(1):14, 2013.

[30] Alejandro Bellogín, Alan Said, and Arjen P de Vries. The magic barrier of recommender systems–no magic, just ratings. In *International Conference*

on User Modeling, Adaptation, and Personalization, pages 25–36. Springer, 2014.

[31] D. Ben-Arieh and Z. Chen. Linguistic labels aggregation and consensus measure for autocratic decision-making using group recommendations. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 36(1):558–568, 2006.

[32] J. Ben Schafer, Joseph A. Konstan, and John Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1-2):115–153, 2001.

[33] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984.

[34] Daniel Billsus and Michael J. Pazzani. User modeling for adaptive news access. *User Modelling and User-Adapted Interaction*, 10(2-3):147–180, 2000.

[35] Yolanda Blanco-Fernandez, Martin Lopez-Nores, Alberto Gil-Solla, Manuel Ramos-Cabrer, and Jose J. Pazos-Arias. Exploring synergies between content-based filtering and Spreading Activation techniques in knowledge-based recommender systems. *Information Sciences*, 181(21):4823–4846, 2011.

[36] Toine Bogers and Antal van den Bosch. Comparing and evaluating information retrieval algorithms for news recommendation. In *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, pages 141–144, New York, NY, USA, 2007. ACM.

[37] Ludovico Boratto and Salvatore Carta. State-of-the-art in group recommendation and new approaches for automatic identification of groups. In

Information retrieval and mining in distributed environments, pages 1–20. Springer, 2010.

[38] Ludovico Boratto and Salvatore Carta. Art: group recommendation approaches for automatically detected groups. *International Journal of Machine Learning and Cybernetics*, 6(6):953–980, 2015.

[39] J.C. Borda. Memorie sur les elections au scrutin. *Historie de l'Academie Royale des Sciences (Paris)*, 1781.

[40] G. Bordogna, M. Fedrizzi, and G. Pasi. A linguistic modeling of consensus in group decision making based on OWA operators. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 27(1):126–133, 1997.

[41] 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, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[42] Derek Bridge and Alex Ferguson. An expressive query language for product recommender systems. *Artificial Intelligence Review*, 18(3-4):269–307, 2002.

[43] Robin Burke. Integrating Knowledge-based and Collaborative-filtering Recommender Systems. In *Workshop on AI and Electronic Commerce*, 1999.

[44] Robin Burke. Knowledge-based recommender systems. *Encyclopedia of Library and Information Systems*, 69(32):175–186, 2000.

[45] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modelling and User-Adapted Interaction*, 12(4):331–370, 2002.

[46] Robin Burke. Interactive critiquing for catalog navigation in E-commerce. *Artificial Intelligence Review*, 18(3-4):245–267, 2002.

[47] Robin Burke. Hybrid web recommender systems. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The adaptive web*, pages 377–408. Springer-Verlag, 2007.

[48] Robin Burke, MichaelP. O'Mahony, and NeilJ. Hurley. Robust collaborative recommendation. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 961–995. Springer US, 2015.

[49] C.T.L. Butler and A. Rothstein. *On Conflict and Consensus: A Handbook on Formal Consensus Decision Making*. Food Not Bombs Publishing, 2006.

[50] Laurent Candillier, Max Chevalier, Damien Dudognon, and Josiane Mothe. Diversity in recommender systems. In *Proceedings of the 4th International Conference on Advances in Human-Oriented and Personalized Mechanisms, Technologies and Services (CENTRIC)*, pages 48–53, 2011.

[51] Iván Cantador and Pablo Castells. *Recommender Systems for the Social Web*, chapter Group Recommender Systems: New Perspectives in the Social Web, pages 139–157. Springer Berlin Heidelberg, 2012.

[52] P. Casagranda, M. L. Sapino, and K. S. Candan. Audio assisted group detection using smartphones. In *Multimedia Expo Workshops (ICMEW), 2015 IEEE International Conference on*, pages 1–6, June 2015.

[53] Giovanna Castellano, Anna Maria Fanelli, and Maria Alessandra Torsello. Newer: A system for neuro-fuzzy web recommendation. *Applied Soft Computing*, 11(1):793–806, 2011.

[54] Jorge Castro, Rosa M. Rodríguez, and Manuel J. Barranco. Weighting of features in content-based filtering with entropy and dependence measures. *International Journal of Computational Intelligence Systems*, 7(1):80–89, 2014.

[55] Roberto Centeno, Ramón Hermoso, and Maria Fasli. On the inaccuracy of numerical ratings: dealing with biased opinions in social networks. *Information Systems Frontiers*, 17(4):809–825, 2015.

[56] Li Chen and Pearl Pu. Interaction design guidelines on critiquing-based recommender systems. *User Modelling and User-Adapted Interaction*, 19(3):167–206, 2009.

[57] S. M. Chen and B. H. Tsai. Autocratic decision making using group recommendations based on intervals of linguistic terms and likelihood-based comparison relations. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(2):250–259, 2015.

[58] F. Chiclana, E. Herrera-Viedma, S. Alonso, and R.A. Marqués-Pereira. *H. Bustince et al. (eds.), Fuzzy Sets and Their Extensions: Representation, Aggregation and Models*, chapter Preferences and Consistency Issues in Group Decision Making, pages 219–237. Springer, 2008.

[59] Luis Omar Colombo-Mendoza, Rafael Valencia-García, Alejandro Rodríguez-González, Giner Alor-Hernández, and José Javier Samper-Zapater. Recommetz: A context-aware knowledge-based mobile recommender system for movie showtimes. *Expert Systems with Applications*, 42(3):1202–1222, 2015.

[60] Olivier Coutand, Olaf Drögehorn, Klaus David, Petteri Nurmi, Patrik Floreen, Ralf Kernchen, Silke Holtmanns, Stefano Campadello, Theo Kanter, Miquel Martin, Ronald van Eijk, and Renate Guarneri. Context-aware group management in mobile environments. In *Proceedings of the 14th IST Mobile & Wireless Communications Summit*, pages 1–6, June 2005.

[61] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *The fourth ACM Conference on Recommender Systems*, pages 39–46, 2010.

[62] Nuno Crokidakis, Victor H Blanco, and Celia Anteneodo. Impact of contrarians and intransigents in a kinetic model of opinion dynamics. *Physical Review E*, 89(1):013310, 2014.

[63] Marco de Gemmis, Pasquale Lops, Cataldo Musto, Fedelucio Narducci, and Giovanni Semeraro. Semantics-aware content-based recommender systems. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 119–159. Springer US, 2015.

[64] Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. *User Modeling and User-Adapted Interaction*, 17(3):217–255, 2007.

[65] Carmen De Maio, Giuseppe Fenza, Matteo Gaeta, Vincenzo Loia, Francesco Orciuoli, and Sabrina Senatore. Rss-based e-learning recommendations exploiting fuzzy fca for knowledge modeling. *Applied Soft Computing*, 12(1):113–124, 2012.

[66] Toon De Pessemier, Cédric Courtois, Kris Vanhecke, Kristin Van Damme, Luc Martens, and Lieven De Marez. A user-centric evaluation of context-aware recommendations for a mobile news service. *Multimedia Tools and Applications*, 75(6):3323–3351, Mar 2016.

[67] Toon De Pessemier, Simon Dooms, and Luc Martens. Comparison of group recommendation algorithms. *Multimed. Tools. Appl.*, 72(3):2497–2541, 2014.

[68] Morris H DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

[69] Yucheng Dong, Xia Chen, Haiming Liang, and Cong-Cong Li. Dynamics of linguistic opinion formation in bounded confidence model. *Information Fusion*, 32:52–61, 2016.

[70] Yucheng Dong, Zhaogang Ding, Luis Martínez, and Francisco Herrera. Managing consensus based on leadership in opinion dynamics. *Information Sciences*, 397:187–205, 2017.

[71] Simon Dooms, Toon De Pessemier, and Luc Martens. Movietweetings: a movie rating dataset collected from twitter. In *Workshop on Crowdsourcing and Human Computation for Recommender Systems, CrowdRec at RecSys 2013*, 2013.

[72] E. B. dos Santos, A. F. da Costa, R. M. D'addio, M. G. Manzato, and R. Goularte. Introducing the concept of always-welcome recommendations. In *Computer and Information Science (ICIS), 2015 IEEE/ACIS 14th International Conference on*, pages 197–202, June 2015.

[73] Pragya Dwivedi and Kamal K Bharadwaj. e-learning recommender system for a group of learners based on the unified learner profile approach. *Expert Systems*, 32:264 – 276, 2015.

[74] Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622. ACM, 2001.

[75] Michael D Ekstrand, John T Riedl, and Joseph A Konstan. Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction*, 4(2):81–173, 2011.

[76] Ugo Erra, Sabrina Senatore, Fernando Minnella, and Giuseppe Caggianese. Approximate tf–idf based on topic extraction from massive message stream using the gpu. *Information Sciences*, 292:143 – 161, 2015.

[77] Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, and Markus Zanker. An integrated environment for the development of knowledge-based recommender applications. *International Journal of Electronic Commerce*, 11(2):11–34, 2006.

[78] Alexander Felfernig, Christoph Zehentner, Gerald Ninaus, Harald Grabner, Walid Maalej, Dennis Pagano, Leopold Weninger, and Florian Reinfrank. *Group Decision Support for Requirements Negotiation*, pages 105–116. Springer Berlin Heidelberg, 2012.

[79] Alejandro Figueroa and Günter Neumann. Context-aware semantic classification of search queries for browsing community question–answering archives. *Knowledge-Based Systems*, 96:1 – 13, 2016.

[80] Daniel M. Fleder and Kartik Hosanagar. Recommender systems and their impact on sales diversity. In *Proceedings of the 8th ACM Conference on Electronic Commerce*, pages 192–199, New York, NY, USA, 2007. ACM.

[81] Francois Fouss, Youssef Achbany, and Marco Saerens. A probabilistic reputation model based on transaction ratings. *Information Sciences*, 180(11):2095–2123, JUN 1 2010.

[82] Noah E Friedkin and Eugene C Johnsen. *Social influence network theory: A sociological examination of small group dynamics*, volume 33. Cambridge University Press, 2011.

[83] Simon Funk. Netflix update: Try this at home, 2006.

[84] R.B. Gallupe. The tyranny of methodologies in information systems research. *SIGMIS Database*, 38(3):20–28, 2007.

[85] Enrique Garcia, Cristobal Romero, Sebastian Ventura, and Carlos de Castro. An architecture for making recommendations to courseware authors using association rule mining and collaborative filtering. *User Modelling and User-Adapted Interaction*, 19(1-2):99–132, FEB 2009.

[86] Inma Garcia, Sergio Pajares, Laura Sebastia, and Eva Onaindia. Preference elicitation techniques for group recommender systems. *Information Sciences*, 189:155 – 175, 2012.

[87] Mike Gartrell, Xinyu Xing, Qin Lv, Aaron Beach, Richard Han, Shivakant Mishra, and Karim Seada. Enhancing group recommendation by incorporating social relationship interactions. In *Proceedings of the 16th ACM International Conference on Supporting Group Work*, GROUP '10, pages 97–106, New York, NY, USA, 2010. ACM.

[88] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, and Grammati Pantziou. Mobile recommender systems in tourism. *Journal of Network and Computer Applications*, 39(0):319 – 333, 2014.

[89] Y. Ghanghas, C. Rana, and S. Dhingra. Diversity in recommender system. *International Journal of Engineering Trends and Technology*, 4:2344–2348, 2013.

[90] Mustansar Ghazanfar and Adam Prugel-Bennett. An improved switching hybrid recommender system using naive bayes classifier and collaborative filtering. In *The 2010 IAENG International Conference on Data Mining and Applications*, 2010.

[91] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Interacting with Computers of the ACM*, 35:61–70, 1992.

[92] T. González-Arteaga, J.C.R. Alcantud, and R. de Andrés Calle. New correlation coefficients for hesitant fuzzy sets. In *The 16th World Congress of the International Fuzzy Systems Association*, pages 427–434, 2015.

[93] Michele Gorgoglione, Umberto Panniello, and Alexander Tuzhilin. The effect of context-aware recommendations on customer purchasing behavior and trust. In *Proc. of the fifth ACM conference on RS*, RecSys '11, pages 85–92, New York, USA, 2011. ACM.

[94] Jagadeesh Gorla, Neal Lathia, Stephen Robertson, and Jun Wang. Probabilistic group recommendation via information matching. In *Proceedings*

of the 22nd international conference on World Wide Web, pages 495–504. International World Wide Web Conferences Steering Committee, 2013.

[95] Asela Gunawardana and Guy Shani. *Evaluating recommender systems*, pages 265–308. Springer US, 2015.

[96] Ihsan Gunes, Cihan Kaleli, Alper Bilge, and Huseyin Polat. Shilling attacks against recommender systems: a comprehensive survey. *Artificial Intelligence Review*, 42(4):767–799, 2014.

[97] B. Guo, Z. Yu, L. Chen, X. Zhou, and X. Ma. Mobigroup: Enabling lifecycle support to social activity organization and suggestion with mobile crowd sensing. *IEEE Transactions on Human-Machine Systems*, PP(99):1–13, 2015.

[98] J. Guo, Y. Zhu, A. Li, Q. Wang, and W. Han. A social influence approach for group user modeling in group recommendation systems. *IEEE Intelligent Systems*, PP(99):1–1, 2016.

[99] Xuetao Guo and Jie Lu. Intelligent e-government services with personalized recommendation techniques. *International Journal of Intelligent Systems*, 22(5):401–417, 2007.

[100] Ido Guy, Inbal Ronen, Elad Kravi, and Maya Barnea. Increasing activity in enterprise online communities using content recommendation. *ACM Trans. Comput.-Hum. Interact.*, 23(4):22:1–22:28, 2016.

[101] Byeong-Jun Han, Seungmin Rho, Sanghoon Jun, and Eenjun Hwang. Music emotion classification and context-based music recommendation. *Multimedia Tools and Applications*, 47(3):433–460, 2010.

[102] Rainer Hegselmann, Ulrich Krause, et al. Opinion dynamics and bounded confidence models, analysis, and simulation. *Journal of Artificial Societies and Social Simulation*, 5(3), 2002.

[103] JL Herlocker, JA Konstan, K Terveen, and JT Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, JAN 2004.

[104] Jon Herlocker, Joseph A. Konstan, and John Riedl. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, 5(4):287–310, 2002.

[105] Alfred Hermida. Twittering the news. *Journalism Practice*, 4(3):297–308, 2010.

[106] F. Herrera, E. Herrera-Viedma, and J.L. Verdegay. A sequential selection process in group decision making with linguistic assessments. *Information Sciences*, 85(4):223–239, 1995.

[107] F. Herrera, E. Herrera-Viedma, and J.L. Verdegay. A model of consensus in group decision making under linguistic assessments. *Fuzzy sets and Systems*, 78(1):73–87, 1996.

[108] F. Herrera, E. Herrera-Viedma, and J.L. Verdegay. Linguistic measures based on fuzzy coincidence for reaching consensus in group decision making. *International Journal of Approximate Reasoning*, 16(3-4):309–334, 1997.

[109] E. Herrera-Viedma, J.L. García-Lapresta, J. Kacprzyk, M. Fedrizzi, H. Nurmi, S. Zadrozny, and (Eds.). *Consensual Processes*, volume 267 of *Studies in Fuzziness and Soft Computing*. Springer-Verlag, 2011.

[110] Enrique Herrera-Viedma, Francisco Javier Cabrerizo, Janusz Kacprzyk, and Witold Pedrycz. A review of soft consensus models in a fuzzy environment. *Information Fusion*, 17:4–13, 2014.

[111] M. Anwar Hossain, Jorge Parra, Pradeep K. Atrey, and Abdulmotaleb Saddik. A framework for human-centered provisioning of ambient media services. *Multimedia Tools and Applications*, 44(3):407–431, 2009.

[112] Tim Hussein, Timm Linder, Werner Gaulke, and Juergen Ziegler. A framework and an architecture for context-aware group recommendations. In *Proceedings of Collaboration and Technology: 16th International Conference, CRIWG 2010*, pages 121–128. Springer Berlin Heidelberg, 2010.

[113] Tim Hussein, Timm Linder, Werner Gaulke, and Jürgen Ziegler. Hybreed: A software framework for developing context-aware hybrid recommender systems. *User Modeling and User-Adapted Interaction*, 24(1-2):121–174, 2014.

[114] Anthony Jameson and Barry Smyth. Recommendation to groups. In *The adaptive web*, pages 596–627. Springer, 2007.

[115] Dietmar Jannach. Fast computation of query relaxations for knowledge-based recommenders. *AI Interacting with Computers*, 22(4):235–248, 2009.

[116] Dietmar Jannach and Johannes Liegl. Conflict-directed relaxation of constraints in content-based recommender systems. In *Proceedings of the 19th International Conference on Advances in Applied Artificial Intelligence: Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 819–829, Berlin, Heidelberg, 2006. Springer-Verlag.

[117] Martijn Kagie, Michiel Wezel, and Patrick J.F. Groenen. Map based visualization of product catalogs. In *Recommender Systems Handbook*, chapter 17, pages 547–576. Springer US, 2011.

[118] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse recommendation: N-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the Fourth ACM*

Conference on Recommender Systems, RecSys '10, pages 79–86, New York, NY, USA, 2010. ACM.

[119] O. Khalid, M. U. S. Khan, S. U. Khan, and A. Y. Zomaya. Omnisuggest: A ubiquitous cloud-based context-aware recommendation system for mobile social networks. *IEEE Transactions on Services Computing*, 7(3):401–414, July 2014.

[120] Daniel Kluver, Tien T Nguyen, Michael Ekstrand, Shilad Sen, and John Riedl. How many bits per rating? In *Proceedings of the sixth ACM conference on Recommender systems*, pages 99–106. ACM, 2012.

[121] T. Kohonen. *Self-organizing maps*. Heidelberg: Springer, 1995.

[122] Joseph A Konstan and John Riedl. Recommender systems: from algorithms to user experience. *User Modeling and User-Adapted Interaction*, 22(1-2):101–123, 2012.

[123] Yehuda Koren. Collaborative Filtering with Temporal Dynamics. *Interacting with Computers OF THE ACM*, 53(4):89–97, APR 2010.

[124] Yehuda Koren and Robert Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*, chapter 5, pages 145–186. Springer US, 2011.

[125] Yehuda Koren and Robert Bell. *Advances in Collaborative Filtering*, pages 77–118. Springer US, 2015.

[126] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[127] Nicolas Kuchmann-Beauger, Marie-Aude Aufaure, and Raphael Thollot. Context-aware question answering system, 01 13 2015. US Patent 8,935,277.

[128] Ekaterina Kurdyukova, Stephan Hammer, and Elisabeth André. Personalization of content on public displays driven by the recognition of group context. In *Proceedings of the Third International Joint Conference on Ambient Intelligence, AmI 2012, Pisa, Italy*, pages 272–287. Springer Berlin Heidelberg, 2012.

[129] Aristomenis S. Lampropoulos, Paraskevi S. Lampropoulou, and George A. Tsihrintzis. A cascade-hybrid music recommender system for mobile services based on musical genre classification and personality diagnosis. *Multimedia Tools and Applications*, 59(1):241–258, 2012.

[130] Bin Li, Ling Chen, Xingquan Zhu, and Chengqi Zhang. Noisy but non-malicious user detection in social recommender systems. *World Wide Web*, 16(5-6):677–699, 2013.

[131] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073, 2014.

[132] S. Loeb and E. Panagos. Information filtering and personalization: Context, serendipity and group profile effects. In *2011 IEEE Consumer Communications and Networking Conference (CCNC)*, pages 393–398, Jan 2011.

[133] Pasquale Lops, Marco Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*, chapter 3, pages 73–105. Springer US, 2011.

[134] Jan Lorenz. Continuous opinion dynamics under bounded confidence: A survey. *International Journal of Modern Physics C*, 18(12):1819–1838, 2007.

[135] J. Lu, G. Zhang, D. Ruan, and F. Wu. *Multi-Objective Group Decision Making*. Imperial College Press, 2006.

[136] Jie Lu, Qusai Shambour, Yisi Xu, Qing Lin, and Guangquan Zhang. A web-based personalized business partner recommendation system using fuzzy semantic techniques. *Computational Intelligence*, 29(1):37–69, 2013.

[137] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender system application developments: A survey. *Decision Support Systems*, 74:12–32, 2015.

[138] N. Manouselis, R. Vuorikari, and F. Van Assche. Collaborative recommendation of e-learning resources: an experimental investigation. *Journal of Computer Assisted Learning*, 26(4):227–242, 2010.

[139] L. Martínez, L.G. Pérez, M. Barranco, L.G. Pérez, and M. Espinilla. Improving the effectiveness of knowledge based recommender systems using incomplete linguistic preference relations. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 16(Supplement-2):33–56, 2008.

[140] L. Martínez, R. M. Rodríguez, and M. Espinilla. Reja: A georeferenced hybrid recommender system for restaurants. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 03*, WI-IAT '09, pages 187–190. IEEE Computer Society, 2009.

[141] Luis Martínez, Manuel J. Barranco, Luis G. Pérez, and Macarena Espinilla. A knowledge based recommender system with multigranular linguistic information. *International Journal of Computational Inteligence Systems*, 1(3):225–236, 2008.

[142] P Massa and P Avesani. Trust-aware collaborative filtering for recommender systems. In Meersman, R and Tari, Z and VanderAalst, W and Bussler, C and Gal, A and Cahill, V and Vinoski, S and Vogels, W and Gatarci, T and Sycara, K, editor, *ON THE MOVE TO MEANINGFUL INTERNET*

SYSTEMS 2004: COOPIS, DOA, AND ODBASE, PT 1, PROCEEDINGS, volume 3290 of *Lecture Notes in Computer Science*, pages 492–508. RMIT Univ, Sch Comp Sci & Informat Technol; Vrije Univ Brussel, Dept Comp Sci, Springer-Verlag Berlin, 2004.

[143] Judith Masthoff. Group recommender systems: Combining individual models. In *Recommender Systems Handbook*, chapter 21, pages 677–702. Springer US, 2011.

[144] Judith Masthoff. Group recommender systems: Aggregation, satisfaction and group attributes. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 743–776. Springer US, 2015.

[145] Judith Masthoff and Albert Gatt. In pursuit of satisfaction and the prevention of embarrassment: affective state in group recommender systems. *User Modelling and User-Adapted Interaction*, 16(3-4):281–319, SEP 2006.

[146] Joseph F. McCarthy and Theodore D. Anagnost. Musicfx: an arbiter of group preferences for computer supported collaborative workouts. In *Proc. of CSCW '98*, pages 363–372. ACM, 1998.

[147] Kevin McCarthy, Maria Salamó, Lorcan Coyle, Lorraine McGinty, Barry Smyth, and Paddy Nixon. Cats: A synchronous approach to collaborative group recommendation. In *FLAIRS Conference*, volume 2006, pages 86–91, 2006.

[148] Prem Melville, Raymond J Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. *Aaai/iaai*, 23:187–192, 2002.

[149] Christos Mettouris and George A Papadopoulos. Contextual modelling in context-aware recommender systems: a generic approach. In *Web Informa-*

tion Systems Engineering–WISE 2011 and 2012 Workshops, pages 41–52. Springer, 2013.

[150] Christos Mettouris and George A Papadopoulos. Designing context models for cars incorporating partially observable context. In *Modeling and Using Context*, pages 118–131. Springer, 2015.

[151] Catarina Miranda and Alípio Mário Jorge. *Proceedings of Progress in Artificial Intelligence: 14th Portuguese Conference on Artificial Intelligence.*, chapter Item-Based and User-Based Incremental Collaborative Filtering for Web Recommendations, pages 673–684. Springer Berlin Heidelberg, 2009.

[152] Bamshad Mobasher, Robert Cooley, and Jaideep Srivastava. Automatic personalization based on web usage mining. *Communications of the ACM*, 43(8):142–151, 08 2000.

[153] Raymond J. Mooney and Loriene Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, DL '00, pages 195–204, New York, NY, USA, 2000. ACM.

[154] Cataldo Musto, Giovanni Semeraro, Pasquale Lops, Marco de Gemmis, and Georgios Lekkas. Personalized finance advisory through case-based recommender systems and diversification strategies. *Decision Support Systems*, 77:100 – 111, 2015.

[155] Krzysztof Myszkorowski and Danuta Zakrzewska. Using fuzzy logic for recommending groups in e-learning systems. In *Computational Collective Intelligence. Technologies and Applications*, pages 671–680. Springer, 2013.

[156] Krzysztof Myszkorowski and Danuta Zakrzewska. Building contextual student group recommendations with fuzzy logic. In *Rough Sets and Current Trends in Computing*, pages 358–365. Springer, 2014.

[157] T. T. S. Nguyen, H. Y. Lu, and J. Lu. Web-page recommendation based on web usage and domain knowledge. *IEEE Transactions on Knowledge and Data Engineering*, 26(10):2574–2587, 2014.

[158] Mehrbakhsh Nilashi, Othman Ibrahim, and Karamollah Bagherifard. A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques. *Expert Systems with Applications*, 92:507 – 520, 2018.

[159] Xia Ning, Christian Desrosiers, and George Karypis. A comprehensive survey of neighborhood-based recommendation methods. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 37–76. Springer US, 2015.

[160] Li Niu. *The cognition-driven decision process for business intelligence: a model and techniques*. PhD thesis, University of Technology Sydney, 2009.

[161] J.M. Noguera, M.J. Barranco, R.J. Segura, and L. Martínez. A mobile 3d-gis hybrid recommender system for tourism. *Information Sciences*, 215:37–52, 2012.

[162] Mark O'Connor, Dan Cosley, Joseph A. Konstan, and John Riedl. Polylens: a recommender system for groups of users. In *Proceedings ECSCW'01*, pages 199–218, 2001.

[163] Obaro Odiete, Tanvi Jain, Ifeoma Adaji, Julita Vassileva, and Ralph Deters. Recommending programming languages by identifying skill gaps using analysis of experts. a study of stack overflow. In *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization*, UMAP '17, pages 159–164, New York, NY, USA, 2017. ACM.

[164] Michael P O'Mahony, Neil J Hurley, and Guénolé Silvestre. Detecting noise in recommender system databases. In *Proceedings of the 11th international conference on Intelligent user interfaces*, pages 109–115. ACM, 2006.

[165] Michael P. O'Mahony and Barry Smyth. Collaborative web search: a robustness analysis. *Artificial Intelligence Review*, 28(1):69–86, JUN 2007.

[166] S.A. Orlovsky. Decision-making with a fuzzy preference relation. *Fuzzy Sets and Systems*, 1(3):155–167, July 1978.

[167] Fernando Ortega, Antonio Hernando, Jesus Bobadilla, and Jeon Hyung Kang. Recommending items to group of users using matrix factorization based collaborative filtering. *Information Sciences*, 345:313 – 324, 2016.

[168] I. Palomares, F.J. Estrella, L. Martínez, and F. Herrera. Consensus under a fuzzy context: Taxonomy, analysis framework AFRYCA and experimental case of study. *Information Fusion*, 20(November 2014):252–271, 2014.

[169] I. Palomares, J. Liu, Y. Xu, and L. Martínez. Modelling experts' attitudes in group decision making. *Soft Computing*, 16(10):1755–1766, 2012.

[170] I. Palomares and L. Martínez. A semi-supervised multi-agent system model to support consensus reaching processes. *IEEE Transactions on Fuzzy Systems*, 22(4):762 – 777, 2014.

[171] I. Palomares, L. Martínez, and F. Herrera. MENTOR: A graphical monitoring tool of preferences evolution in large-scale group decision making. *Knowledge-based Systems*, 58 (Spec. Iss. Intelligent Decision Support Making Tools and Techniques):66–74, 2014.

[172] Umberto Panniello and Michele Gorgoglione. Incorporating context into recommender systems: an empirical comparison of context-based approaches. *Electronic Commerce Research*, 12(1):1–30, 2012.

[173] Umberto Panniello, Alexander Tuzhilin, and Michele Gorgoglione. Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction*, 24(1-2):35–65, 02 2014.

[174] Nish Parikh and Neel Sundaresan. Buzz-based recommender system. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 1231–1232, New York, NY, USA, 2009. ACM.

[175] R.O. Parreiras, P. Ekel, and F. Bernardes Jr. A dynamic consensus scheme based on a nonreciprocal fuzzy preference relation modeling. *Information Sciences*, 211(1):1–17, 2012.

[176] Michael J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5-6):393–408, 1999.

[177] M.J. Pazzani and D. Billsus. Content-Based Recommendation Systems. *The Adaptive Web*, 4321:325–341, 2007.

[178] K. Pearson. Notes on the history of correlation. *Biometrika*, 13(1):25–45, 1920.

[179] Hau Xuan Pham and Jason J Jung. Preference-based user rating correction process for interactive recommendation systems. *Multimedia tools and applications*, 65(1):119–132, 2013.

[180] Parivash Pirasteh, Dosam Hwang, and Jason J Jung. Exploiting matrix factorization to asymmetric user similarities in recommendation systems. *Knowledge-Based Systems*, 83:51–57, 2015.

[181] L. Ponzanelli, G. Bavota, M. D. Penta, R. Oliveto, and M. Lanza. Prompter: A self-confident recommender system. In *2014 IEEE International Conference on Software Maintenance and Evolution*, pages 577–580, Sept 2014.

[182] Luca Ponzanelli. Holistic recommender systems for software engineering. In *Companion Proceedings of the 36th International Conference on Software Engineering*, ICSE Companion 2014, pages 686–689, New York, NY, USA, 2014. ACM.

[183] Luca Ponzanelli, Simone Scalabrino, Gabriele Bavota, Andrea Mocci, Rocco Oliveto, Massimiliano Di Penta, and Michele Lanza. Supporting software developers with a holistic recommender system. In *Proceedings of the 39th International Conference on Software Engineering*, ICSE '17, pages 94–105, Piscataway, NJ, USA, 2017. IEEE Press.

[184] Martin F Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[185] Pearl Pu, Li Chen, and Pratyush Kumar. Evaluating product search and recommender systems for E-commerce environments. *Electronic Commerce Research*, 8(1-2):1–27, 2008.

[186] D. Rafailidis, P. Kefalas, and Y. Manolopoulos. Preference dynamics with multimodal user-item interactions in social media recommendation. *Expert Systems with Applications*, 74:11 – 18, 2017.

[187] D. Rafailidis and A. Nanopoulos. Modeling users preference dynamics and side information in recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(6):782–792, 2016.

[188] James Reilly, Kevin McCarthy, Lorraine McGinty, and Barry Smyth. Dynamic critiquing. In *Advances in Case-Based Reasoning*, volume 3155, pages 763–777. Springer-Verlag Berlin, 2004.

[189] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, pages 175–186, New York, USA, 1994. ACM.

[190] Paul Resnick and Hal R. Varian. Recommender systems. *Communications of the ACM*, 40:56–58, March 1997.

[191] Francesco Ricci. Mobile recommender systems. *International Journal of Information Technology and Tourism*, 12(3):205–231, 2011.

[192] Francesco Ricci, Lior Rokach, and Bracha Shapira. Recommender systems handbook. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, chapter 1, pages 1–35. Springer, 2011.

[193] Francesco Ricci, Lior Rokach, and Bracha Shapira. *Recommender Systems: Introduction and Challenges*, pages 1–34. Springer US, 2015.

[194] J. L. Rodgers and W. A. Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1998.

[195] Rosa M. Rodríguez, Macarena Espinilla, Pedro J. Sánchez, and Luis Martínez-Lopez. Using linguistic incomplete preference relations to cold start recommendations. *Internet Research*, 20(3):296–315, 2010.

[196] L. Roffia, L. Lamorte, G. Zamagni, S. Bartolini, A. D'Elia, F. Spadini, D. Manzaroli, C. A. Licciardi, and T. S. Cinotti. Personalized context based services for dynamic user groups. In *2009 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, pages 411–416, Oct 2009.

[197] Ekkawut Rojsattarat and Nuanwan Soonthornphisaj. Hybrid recommendation: Combining content-based prediction and collaborative filtering. In *Intelligent Data Engineering and Automated Learning*, volume 2690 of *Lecture Notes in Computer Science*, pages 337–344. Springer-Verlag Berlin, 2003.

[198] Alan Said, Shlomo Berkovsky, and Ernesto W. De Luca. Group recommendation in context. In *Proceedings of the 2Nd Challenge on Context-Aware Movie Recommendation*, CAMRa '11, pages 2–4, New York, NY, USA, 2011. ACM.

[199] Alan Said, Brijnesh J. Jain, Sascha Narr, and Till Plumbaum. *Users and Noise: The Magic Barrier of Recommender Systems*, pages 237–248. Springer Berlin Heidelberg, 2012.

[200] S. Saint and J. R. Lawson. *Rules for Reaching Consensus. A Modern Approach to Decision Making*. Jossey-Bass, 1994.

[201] Simone Santini and Ramesh Jain. Similarity measures. *IEEE Transactions on pattern analysis and machine Intelligence*, 21(9):871–883, 1999.

[202] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, Minnesota Univ Minneapolis Dept of Computer Science, 2000.

[203] 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.

[204] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.

[205] Silvia Schiaffino and Analia Amandi. Building an expert travel agent as a software agent. *Expert Systems with Applications*, 36(2, Part 1):1291–1299, 2009.

[206] Christophe Senot, Dimitre Kostadinov, Makram Bouzid, Jérôme Picault, Armen Aghasaryan, and Cédric Bernier. Analysis of strategies for building group profiles. In *18th International Conference on User Modeling, Adaptation, and Personalization*, UMAP'10, pages 40–51, Berlin, Heidelberg, 2010. Springer-Verlag.

[207] Qusai Shambour and Jie Lu. A trust-semantic fusion-based recommendation approach for e-business applications. *Decision Support Systems*, 54(1):768–780, 2012.

[208] Bin Shao and Jiafei Yan. Recommending answerers for stack overflow with lda model. In *Proceedings of the 12th Chinese Conference on Computer Supported Cooperative Work and Social Computing*, ChineseCSCW '17, pages 80–86, New York, NY, USA, 2017. ACM.

[209] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating "word of mouth". In *Proceedings of the SIGCHI Conference on Human factors in Computing Systems*, pages 210–217. ACM Press, 1995.

[210] Simen Fivelstad Smaaberg, Nafiseh Shabib, and John Krogstie. A user-study on context-aware group recommendation for concerts. In *HT (Doctoral Consortium/Late-breaking Results/Workshops)*, 2014.

[211] Barry Smyth. Case-based recommendation. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 342–376. Springer, 2007.

[212] Barry Smyth and Paul Cotter. A personalised tv listings service for the digital tv age. *Knowledge-Based Systems*, 13(2-3):53–59, 2000.

[213] Kostas Stefanidis, Nafiseh Shabib, Kjetil Norvaag, and John Krogstie. *Advances in Conceptual Modeling: ER 2012 Workshops CMS, ECDM-NoCoDA, MoDIC, MORE-BI, RIGiM, SeCoGIS, WISM, Florence, Italy, October 15-18, 2012. Proceedings*, chapter Contextual Recommendations for Groups, pages 89–97. Springer Berlin Heidelberg, 2012.

[214] Ja-Hwung Su, Bo-Wen Wang, Chin-Yuan Hsiao, and Vincent S. Tseng. Personalized rough-set-based recommendation by integrating multiple con-

tents and collaborative information. *Information Sciences*, 180(1):113–131, 2010.

[215] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. Feature-weighted user model for recommender systems. In *Proceedings of the 11th international conference on User Modeling*, pages 97–106. Springer-Verlag, 2007.

[216] G. S. Thyagaraju and U. P. Kulkarni. Interactive democratic group preference algorithm for interactive context aware tv. In *2010 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pages 1–5, Dec 2010.

[217] V. Torra. Hesitant fuzzy sets. *International Journal of Intelligent Systems*, 25(6):529–539, 2010.

[218] V. Torra and Y. Narukawa. On hesitant fuzzy sets and decision. In *Proceedings of the 18th IEEE International Conference on Fuzzy Systems*, pages 1378–1382, Aug 2009.

[219] Shari Trewin. Knowledge-based recommender systems. *Encyclopedia of Library and Information Science: Volume 69-Supplement 32*, 2000.

[220] Vijay K Vaishnavi and William Kuechler. *Design science research methods and patterns: innovating information and communication technology*. Crc Press, 2015.

[221] Saúl Vargas and Pablo Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 109–116, New York, NY, USA, 2011. ACM.

[222] Blanca Vargas-Govea, Gabriel González-Serna, and Rafael Ponce-Medellın. Effects of relevant contextual features in the performance of a restaurant recommender system. *ACM RecSys*, 11(592):56, 2011.

[223] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, and E. Duval. Context-aware recommender systems for learning: A survey and future challenges. *IEEE Transactions on Learning Technologies*, 5(4):318–335, Oct 2012.

[224] Paolo Viappiani, Boi Faltings, and Pearl Pu. Preference-based search using example-critiquing with suggestions. *Journal of Artificial Intelligence Research*, 27:465–503, 2006.

[225] Paolo Viappiani, Pearl Pu, and Boi Faltings. Preference-based search with adaptive recommendations. *AI Interacting with Computers*, 21(2-3):155–175, 2008.

[226] Manolis Vozalis and Konstantinos G Margaritis. Collaborative filtering enhanced by demographic correlation. In *AIAI symposium on professional practice in AI, of the 18th world computer congress*, 2004.

[227] Jing Wang, Hui Li, and Hui Zhao. The contextual group recommendation. In *Intelligent Networking and Collaborative Systems (INCoS), 2013 5th International Conference on*, pages 127–131. IEEE, 2013.

[228] Wei Wang, Guangquan Zhang, and Jie Lu. Member contribution-based group recommender system. *Decision Support Systems*, 87:80 – 93, 2016.

[229] Yuanyuan Wang, Stephen Chi-Fai Chan, and Grace Ngai. Applicability of demographic recommender system to tourist attractions: A case study on trip advisor. In *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Volume 03*, WI-IAT '12, pages 97–101. IEEE Computer Society, 2012.

[230] D. Wu, J. Lu, and G. Zhang. A fuzzy tree matching-based personalized e-learning recommender system. *IEEE Transactions on Fuzzy Systems*, 23(6):2412–2426, 2015.

[231] Z. Wu and J. Xu. A consistency and consensus based decision support model for group decision making with multiplicative preference relations. *Decision Support Systems*, 52(3):757–767, 2012.

[232] Huadong Xia, Jiangzhuo Chen, Madhav Marathe, and Henning Mortveit. Synthesis and refinement of detailed subnetworks in a social contact network for epidemic simulations. In John Salerno, Shanchieh Yang, Dana Nau, and Sun-Ki Chai, editors, *Social Computing, Behavioral-Cultural Modeling and Prediction*, volume 6589 of *Lecture Notes in Computer Science*, pages 366–373. Springer Berlin / Heidelberg, 2011.

[233] NM Xia and ZS Xu. Hesitant fuzzy information aggregation in decision making. *International Journal of Approximate Reasoning*, 52:395–407, 2011.

[234] Bo Xiao and Izak Benbasat. E-commerce product recommendation agents: use, characteristics, and impact. *Management Information Systems Quarterly*, 31(1):137–209, 2007.

[235] Z. Xu. An automatic approach to reaching consensus in multiple attribute group decision making. *Computers & Industrial Engineering*, 56(4):1369–1374, 2009.

[236] J. Xuan, X. Luo, G. Zhang, J. Lu, and Z. Xu. Uncertainty analysis for the keyword system of web events. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(6):829–842, 2016.

[237] Raciel Yera, Jorge Castro, and Luis Martínez. A fuzzy model for managing natural noise in recommender systems. *Applied Soft Computing*, 40:187 – 198, 2016.

[238] Raciel Yera Toledo and Yailé Caballero Mota. An e-learning collaborative filtering approach to suggest problems to solve in programming online judges. *International Journal of Distance Education Technologies*, 12(2):51–65, 2014.

[239] Raciel Yera Toledo, Yailé Caballero Mota, and Luis Martínez. Correcting noisy ratings in collaborative recommender systems. *Knowledge-Based Systems*, 76:96 – 108, 2015.

[240] Lotfi A Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.

[241] Danuta Zakrzewska. Building context-aware group recommendations in e-learning systems. In *Proceedings of the Third International Conference on Computational Collective Intelligence: Technologies and Applications - Volume Part I*, ICCCI'11, pages 132–141, Berlin, Heidelberg, 2011. Springer-Verlag.

[242] Valentina Zanardi and Licia Capra. Dynamic updating of online recommender systems via feed-forward controllers. In *Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 11–19. ACM, 2011.

[243] Markus Zanker. A collaborative constraint-based meta-level recommender. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 139–146, New York, NY, USA, 2008. ACM.

[244] Markus Zanker and Markus Jessenitschnig. Case-studies on exploiting explicit customer requirements in recommender systems. *User Modelling and User-Adapted Interaction*, 19(1-2):133–166, 2009.

[245] Markus Zanker, Markus Jessenitschnig, and Wolfgang Schmid. Preference reasoning with soft constraints in constraint-based recommender systems. *Constraints*, 15(4):574–595, 2010.

[246] Cheng Zeng, Dawen Jia, Jian Wang, Liang Hong, Wenhui Nie, Zhihao Li, and Jilei Tian. Context-aware social media recommendation based on potential group. In *Proceedings of the 1st International Workshop on Context Discovery and Data Mining*, ContextDD '12, pages 5:1–5:9, New York, NY, USA, 2012. ACM.

[247] G. Zhang, Y. Dong, Y. Xu, and H. Li. Minimum-cost consensus models under aggregation operators. *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, 41(6):1253–1261, 2011.

[248] Xiaoying Zhang, Junzhou Zhao, and John Lui. Modeling the assimilation-contrast effects in online product rating systems: Debiasing and recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 98–106. ACM, 2017.

[249] Zi-Ke Zhang, Chuang Liu, Yi-Cheng Zhang, and Tao Zhou. Solving the cold-start problem in recommender systems with social tags. *EPL (Europhysics Letters)*, 92(2):28002, 2010.

[250] Laijun Zhao, Xiaoyan Qiu, Xiaoli Wang, and Jiajia Wang. Rumor spreading model considering forgetting and remembering mechanisms in inhomogeneous networks. *Physica A: Statistical Mechanics and its Applications*, 392(4):987–994, 2013.

[251] Nan Zheng and Hongyun Bao. Flickr group recommendation based on user-generated tags and social relations via topic model. In *Advances in Neural Networks–ISNN 2013*, pages 514–523. Springer, 2013.

[252] X. L. Zheng, C. C. Chen, J. L. Hung, W. He, F. X. Hong, and Z. Lin. A hybrid trust-based recommender system for online communities of practice. *IEEE Transactions on Learning Technologies*, 8(4):345–356, Oct 2015.

[253] Yong Zheng. Improve general contextual slim recommendation algorithms by factorizing contexts. In *Proceedings of the 30th Annual ACM Symposium*

on Applied Computing, SAC '15, pages 929–930, New York, NY, USA, 2015. ACM.

[254] Yong Zheng, Bamshad Mobasher, and Robin Burke. Deviation-based contextual slim recommenders. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 271–280, New York, NY, USA, 2014. ACM.

[255] Tao Zhou, Zoltan Kuscsik, Jian-Guo Liu, Matus Medo, Joseph Rushton Wakeling, and Yi-Cheng Zhang. Solving the apparent diversity-accuracy dilemma of recommender systems. *PROCEEDINGS OF THE NATIONAL ACADEMY OF SCIENCES OF THE UNITED STATES OF AMERICA*, 107(10):4511–4515, MAR 9 2010.

[256] C.N. Ziegler, S.M. McNee, J.A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *14th Internation Conference on World Wide Web, Chiba, Japan*, pages 22–32, 2005.

Appendix A

Searching groups without consensus.

This appendix details how to build groups of users that do not reach consensus based on a given similarity measure between users. This method is used in the experiment shown in Section 5.3.5 to measure the effect of the consensus in the recommendation results.

To analyze if a group reaches consensus following DeGroot model, the presence/absence of opinion leaders is determined. If the group has at least one opinion leader, then the group reaches the consensus. Therefore, to find groups that do not reach consensus we need to find groups groups without opinion leaders.

The number of different subsets of size k over a set of size n is $\binom{n}{k}$. Given that in RSs the number of users is large, the number of possible groups is large. For example, the versions of Movielens dataset are 100k, 1m, 10m, 20m and have 943, 6040, 69878, and 138494 users, respectively. Netflix dataset has 480189 users. Therefore, exhaustive search is not an option to find groups without opinions opinion leaders of arbitrary size k.

Additionally, the way in which the weighting matrix is computed and the features of the data affect also to the density of groups that fulfill the restriction compared to the number of combinations. With the definition done (see Eq. B.5), the number of possible groups of size k is too large compared to the number of groups that satisfy the restriction, therefore a depth-first search with random starting point

is performed.

If we analyze the restriction made over the group, $E_G^{leaders} = \emptyset$, to build groups fulfilling this restriction, it is clear that we need to analyze the way in which the pairwise relationships are computed.

If a given group G does not have opinion leaders, then the addition of a user $G \cup \{u\}$ might result in a group with opinion leaders, if the user being added has relationships with the opinion leaders of the subgroup. On the other hand, if the user being added only has relationships with the members of one network partition, the group will have no opinion leaders.

On the other hand, if a group G has at least one opinion leader, the addition of a user u could either maintain or break this situation. The resulting group $G \cup \{u\}$ has opinion leaders if the user u is connected to the group. If the user u does not have connections to the group, or the connections are from opinion followers to user u, then the resulting group will have no opinion leaders.

Once we have analyzed the restriction, we can focus on maintaining the absence of opinion leaders. The implementation of the algorithm is based on the fact that a group without opinion leaders can be decomposed in a series of user additions that fulfill the restriction. To allow this construction, a group of one member is considered to have no opinion leader. Then, members are added to the group until it reaches the desired size maintaining the restriction of having no opinion leader. This way, we can build groups with no opinion leaders inductively.

This decomposition is illustrated by an example provided, which is composed of the weighting matrix described in Table A.1 and the decomposition shown in Table A.3.

Table A.1 shows the pairwise similarities of a group of ten users. From this similarity matrix, the weighting matrix can be computed using Equation 2.27. Note that according to DeGroot's model [68] the consensus opinion is a linear combination of the initial opinions of all agents, and the combinational coefficients are related to the left eigenvector associated with the eigenvalue 1 of the weighting

Table A.1: Similarity matrix a group of ten users that have no opinion leaders (rounded to two decimals).

	40	148	260	353	775	792	812	824	866	915
40	1.00	0	0.14	0.43	0.43	0	0.23	0.20	0.31	0.49
148	0	1.00	0	0	0	0.06	0	0	0	0
260	0.21	0	1.00	0.29	0.29	0	0.29	0.17	0.21	0.33
353	0.60	0	0.28	1.00	0.52	0	0.32	0.08	0.20	0.44
775	0.54	0	0.25	0.46	1.00	0	0.18	0.07	0.39	0.50
792	0	0.09	0	0	0	1.00	0	0	0	0
812	0.40	0	0.35	0.40	0.25	0	1.00	0.40	0.10	0.30
824	0.35	0	0.20	0.10	0.10	0	0.40	1.00	0.10	0.25
866	0.55	0	0.25	0.25	0.55	0	0.10	0.10	1.00	0.40
915	0.65	0	0.31	0.42	0.54	0	0.23	0.19	0.31	1.00

Table A.2: Linear combination of the initial opinions for each opinion group (rounded to two decimals).

	λ_{40}	λ_{148}	λ_{260}	λ_{353}	λ_{775}	λ_{792}	λ_{812}	λ_{824}	λ_{866}	λ_{915}
c_{G^1}	0.17	0	0.10	0.13	0.15	0	0.10	0.07	0.10	0.15
c_{G^2}	0	0.59	0	0	0	0.40	0	0	0	0

matrix. The adjacency matrix has two eigenvectors associated to the eigenvalue 1 (see Table A.2), hence group's opinions will polarize to two values regarding the network partition. One partition is composed of users [148,792] and the other is composed of the remaining members.

Table A.3 describes how the group can be described inductively in a series of groups that fulfill the imposed predicate. In every step, the partial result is a group without opinion leaders, and a user is added maintaining the restriction that the group has no opinion leaders. Note that the order in which the users are added from step 3 and afterwards can be permuted without changing the final result.

Data: U,I,R,conditionOverGroup,seed
1 $G = \emptyset$
2 **foreach** *i=1 to N* **do**
3 $seed_i = seed + i$
4 G_i= searchGroup(conditionOverGroup,K,K,$seed_i$)
5 $G = G \cup \{G_i\}$

Algorithm 6: Procedure to search for a given number of groups with the condition imposed.

Algorithm 6 shows the search of the groups with random starting points. The

Table A.3: Decomposition of the group in subgroups without opinion leaders. The user added in each step is marked with the label of the network partition that it belongs to.

	Step										
Users	0	1	2	3	4	5	6	7	8	9	10
40		A	A	A	A	A	A	A	A	A	A
148			B	B	B	B	B	B	B	B	B
260				A	A	A	A	A	A	A	A
353					A	A	A	A	A	A	A
775						A	A	A	A	A	A
792							B	B	B	B	B
812								A	A	A	A
824									A	A	A
866										A	A
915											A

seed determines the ordering in which the users are checked to find groups fulfilling the condition defined by the predicate *conditionOverGroup*. This general searcher relies on a function to find a specific group that has no opinion leaders of the requested size. Given that the method *conditionOverGroup* starts searching from a different point and continues the search in a different order on each cycle of the loop, which is determined by $seed_i$, the method finds groups composed of different users.

Algorithm 7 details how the depth search is done to find a group that fulfills the predicate *conditionOverGroup* in a fast way. The requisite of the predicate is that it must be inductive, this is, a group of arbitrary size k that fulfills the predicate can be decomposed in additions of users whose partial result also fulfills the predicate. The algorithm is implemented following first in-depth search. This way, the recursion allows to rapidly discard the partial results that do not lead to a solution. To do so, the users that can be added are reduced, *getUnexploredUsers* obtains the sub-list of users that have not been explored yet. If getUnexploredUsers returns an empty list or G^k, the search has reached a dead end and the current branch of search is discarded.

The predicate imposed over the groups to be found is described in Algorithm 8. The predicate extracts the weighting matrix of the group and computes the set

```
   Data: U,conditionOverGroup,k,K,seed
1  U_seed = randomSort(U,seed)
2  if (k = 1)then
3      G^1 = {u_i | u_i ∈ U_seed}
4      return G^1
5  else
6      G^{k-1} = searchGroup(conditionOverGroup,k-1,K,seed)
7      G^k = ∅
8      foreach (g in G^{k-1})do
9          U_reduced = getUnexploredUsers(g, U_seed)
10         foreach (u in U_reduced)do
11             if (conditionOverGroup(g ∪ {u}))then
12                 if (k=K)then
13                     return g ∪ {u}
14                 else
15                     G^k = G^k ∪ {g ∪ {u}}
16     return G^k
```

Algorithm 7: Depth search of one group of size k that fulfills the given predicate

of opinion leaders. To do so, it relies on the function *path*, that computes if there is a path to reach from u_j to u_i. Opinion leaders must be accessible from all the remaining members. If the set of opinion leaders is empty, the group fulfills the predicate.

Data: extractSocialNetwork,G
Output: boolean
1 socialNetwork = extractSocialNetwork(G)
2 $G^{leaders} = \emptyset$
3 **foreach** (u_i *in* G)**do**
4 boolean isAccessibleByAll = $true$
5 **foreach** (u_j *in* $G - \{u_i\}$)**do**
6 $isAccessibleByAll = isAccessibleByAll \wedge path(u_j, u_i) \in socialNetwork$
7 **if** ($isAccessibleByAll$)**then**
8 $G^{leaders} = G^{leaders} \cup \{u_i\}$
9 **return** $|G^{leaders}| = 0$

Algorithm 8: Check if a group does not have opinion leaders.

Appendix B

Similarity measures

Many recommendation approaches rely on computing similarities between users, specifically, the user-based collaborative filtering [159, 190] does. In these works, the similarity measure allows to capture the taste of the user and find other users with similar taste, whose ratings are used to predict ratings for unexperienced items.

Definition 7 *A similarity measure is the degree to which two things are similar. It is defined as a real value that expresses how similar two objects are. Their features are given in the same space:*

$$sim : X \times X \rightarrow [0,1] \tag{B.1}$$

Some interesting properties of similarity measures [201] are the following:

- Constancy of self-similarity: $sim(A,A) = sim(B,B)$
- Symmetry: $sim(A,B) = sim(B,A)$
- Transitivity: $sim(A,B) = sim(B,C) = sim(A,C)$
- Maximality: $sim(A,B) \leq sim(A,A)$

In RSs there are some works that take into account asymmetric similarities [180]. Moreover, transitivity is often not hold by RSs similarities because they consider co-rated items, and therefore they potentially use different features to compare two users.

Several similarities have been defined in RSs to compare users regarding various aspects:

- *Similarity of preferences*: They evaluate similarity of taste between the users comparing how they rate items. These measures have been widely explored in the literature to improve the results of the user-based collaborative filtering [159]:

  - *Pearson's correlation coefficient*: Measures the correlation between the ratings given for co-rated items.

$$pcc(u_j,u_k) = \frac{\sum_{i_l \in I_{u_j} \cap I_{u_k}} (r_{u_j i_l} - \overline{r}_{u_j})(r_{u_k i_l} - \overline{r}_{u_k})}{\sqrt{\sum_{i_l \in I_{u_j} \cap I_{u_k}} (r_{u_j i_l} - \overline{r}_{u_j})^2} \sqrt{\sum_{i_l \in I_{u_j} \cap I_{u_k}} (r_{u_k i_l} - \overline{r}_{u_k})^2}} \quad \text{(B.2)}$$

  - *Cosine coefficient*: Measures the similarity of the vectors of co-rated items for two users.

$$cosine(u_j,u_k) = \frac{||\vec{R_{u_j}} * \vec{R_{u_k}}||}{||\vec{R_{u_j}}|| * ||\vec{R_{u_k}}||} \quad \text{(B.3)}$$

- *Demographic similarity*: Measures how demographically similar are the users.

  - Vozalis and Margaritis [226] demographic similarity: Cosine over vector representation of demographic features.

- *Overlapping of experiences*: They evaluate the degree to which users have had similar experiences. These kind of measures are different from rating correlation ones in the sense that they compare if users have been exposed to the same items, without evaluating the user satisfaction towards the experience [180].

  - Relevance factor: Similarity measure often used to correct preference similarities [104]. It is often used as a penalty to similarities computed

with less than certain number of ratings:

$$relevanceFactor(u_j,u_k) = Min(1, \frac{|I_{u_j} \cap I_{u_k}|}{relevanceFactorValue}) \quad \text{(B.4)}$$

– Conditional probability [180]: It measures how much the target profile is covered by the other profile.

$$condProb(u_j,u_k) = \frac{|I_{u_j} \cap I_{u_k}|}{|I_{u_j}|} \quad \text{(B.5)}$$

– Sorensen index [180]: It also takes into account the proportion ratings in common between users:

$$sorensen(u_j,u_k) = \frac{2 * |I_{u_j}| * |I_{u_k}|}{|I_{u_j}| + |I_{u_k}|} \quad \text{(B.6)}$$

- *Combination of several measures*: There are some measures that combine several types of measures:

  – Asymmetric Cosine [180]: Given that measures based on overlapping of experiences discard the value of the rating, which can result in a loss of important information, this measure combines the cosine coefficient, Sorensen index and the conditional probability.

$$asymCos(u_j,u_k) = cosine(u_j,u_k) * condProb(u_j,u_k) * sorensen(u_j,u_k) \quad \text{(B.7)}$$

The explored similarity measures are used in Chapter 5 to extract the interactions between users, which are used to drive the opinion dynamics model.